ADDIS ABABA UNIVERISTY

SCHOOL OF GRADUATE STUDIES

Amharic Question Answering For Definitional, Biographical and Description Questions

Tilahun Abedissa

A THESIS SUBMITTED TO

THE SCHOOL OF GRADUATE STUDIES OF THE ADDIS ABABA UNIVERSITY

IN PARTIAL FULFILLMENT FOR THE DEGREE OF MASTERS OF SCIENCE IN COMPUTER SCIENCE

November, 2013

ADDIS ABABA UNIVERISTY
SCHOOL OF GRADUATE STUDIES
COLLEGE OF NATURAL SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Amharic Question Answering For Definitional, Biographical and Description Questions

Tilahun Abedissa

ADVISOR:

Mulugeta Libsie (PhD)

APPROVED BY

EXAMINING BOARD:

1. Dr. Mulugeta Libsie, Advisor _____
2. _____ _____
3. _____ _____

Dedication

To My Grand Mother

“እማሆይ ዘነበች ተከስተ (እማዬ)”

Acknowledgments

First and foremost, I would like to thank God. St. Mary, St. Michael, and St. Ourael thank you for your blessing and giving me the courage and wisdom to accomplish this thesis.

I would also like to thank my advisor, Dr. Mulugeta Libsie, for his continual support, encouragement, and patience. I greatly appreciate your support from title selection to the end. I am deeply grateful to Desalegn Abebaw for the interesting discussions we had together. Thank you so much for giving me what I needed.

I also wish to express my deepest gratitude to my sister Asgedech Moges and my friend Ephrem Damene, for sharing their precious time and energy in helping me during test question preparation, corpus preparation, and system evaluation. Furthermore, I also would like to thank Mekonen Mulugeta and my collogues at work, for the very friendly and helping atmosphere you offered me in the work place, and I would like to thank Aba HaileGebriel Biru, my classmates, and my friends for all their support and encouragement.

Finally, I would like to thank my family for their continuous love, support and encouragement. It‘s their unselfish care that makes my dream come true. Thank you. God bless you!

Table of Contents

List of Figures

List of Tables

List of Acronyms

API	Application Programming Interface
CLEF	Cross-Language Evaluation Forum
FAQ	Frequently Asked Questions
IR	Information Retrieval
LSA	Latent Semantic Analysis
NER	Named Entity Recognizer
NFQA	Non-Factoid Question Answering
NIST	National Institute of Standards
NLP	Natural Language Processing
NSD	Non-specific Definitional
QA	Question Answering
SVM	Support Vector Machine
TREC	Text REtrieval Conference
VSM	Vector Space Model

Abstract

There are enormous amounts of Amharic text data on the World Wide Web. Since Question Answering (QA) can go beyond the retrieval of relevant documents, it is an option for efficient information access to such text data. The task of QA is to find the accurate and precise answer to a natural language question from a source text. The existing Amharic QA systems handle fact-based questions that usually take named entities as the answers. In this thesis, we focused on a different type of Amharic QA— Amharic non-factoid QA (NFQA) to deal with more complex information needs. The goal of this study is to propose approaches that tackle important problems in Amharic non-factoid QA, specifically in biography, definition, and description questions. The proposed QA system comprises of document preprocessing, question analysis, document analysis, and answer extraction components.

Rule based and machine learning techniques are used for the question classification. The approach in the document analysis component retrieves relevant documents and filters the retrieved documents using filtering patterns for definition and description questions and for biography questions a retrieved document is only retained if it contains all terms in the target in the same order as in the question. The answer extraction component works in type-by-type manner. That is, the definition-description answer extractor extracts sentences using manually crafted answer extraction patterns. The extracted sentences are scored and ranked, and then the answer selection algorithm selects top 5 non-redundant sentences from the candidate answer set. Finally the sentences are ordered to keep their coherence. On the other hand, the biography answer extractor summarizes the filtered documents by merging them, and then the summary is displayed as an answer after it is validated.

We evaluated our QA system in a modular fashion. The n fold cross validation technique is used to evaluate the two techniques utilized in the question classification. The SVM based classifier classifies about 83.3% and the rule based classifier classifies about 98.3% of the test questions correctly. The document retrieval component is tested on two data sets that are analyzed by a stemmer and morphological analyzer. The F-score on the stemmed documents is 0.729 and on the other data it set is 0.764. Moreover, the average F-score of the answer extraction component is 0.592.

Keywords: Amharic definitional, biographical and description question answering, Rule based question classification, SVM based question classification, Document Analysis, Answer Extraction, Answer Selection.

Chapter One: Introduction

1.1 Background

People need precise information to successfully operate in their environment, support their decisions and solve problems. A gap between an individual‘s background knowledge and information needed in a certain context motivates a search for information. However, there is huge amount of electronic information produced by different individuals, organizations, and countries in the world. There are also large amount of Amharic electronic documents. Though getting precise information is not simple, as a result information retrieval (IR) systems/search engines became essential in everyday life as well as in carrying out professional tasks.

IR systems/search engines return results as links which are relevant to a user‘s query and there is a probability of retrieving irrelevant ones. Since they don‘t have the capability of synthesizing user‘s questions and formulate answers [3], users are expected to inspect all the available documents related to the search topic in order to satisfy their needs, which is time consuming and tedious.

The urge to get as concise information as possible for natural language questions using computers, demands the computers to have the capability of understanding of natural language questions. Question answering (QA) is proposed to address this problem. In contrast with traditional document retrieval systems/search engines, which return ranked lists of potentially relevant documents that users must then manually browse through, question answering systems attempt to directly provide users one or more concise answers in the form of sentences or phrases to natural language questions [1]. Also a broad range of information needs can often be stated as a question.

Question answering can also be interpreted as a sub-discipline of IR with added challenge of applying sophisticated techniques to identify the complex syntactic and semantic relationships present in texts. Also it is widely accepted that Question Answering represents a step beyond standard information retrieval, allowing a more sophisticated and satisfactory response to user‘s information needs [2].

Most current work on QA, which has been rekindled largely by the TREC Text Retrieval Conference and by the cross-lingual QA Track at CLEF (Cross-Language Evaluation Forum*)*, is

somewhat different in nature from querying structured data. These evaluation campaigns foster research in QA from the IR perspective, where the task consists in finding the text that contains the answer to the question and extracting the answer [8, 9]. Moreover most question answering research has at its core a standard pipeline QA system [3, 4] that combines several components in a sequential fashion. Such question answering systems include components corresponding to the following stages in the question answering process:

- Question Analysis – the stage in which questions are processed (e.g., part of speech tagging, named entity extraction, parsing), analyzed, and classified. It also attempts to identify the structure and type of the expected answer.
- Information/Document Retrieval – the stage in which queries are formulated according to question types, question keywords, and additional content. Based on these queries, relevant documents or passages likely to contain correct answers are retrieved.
- Answer Extraction – the stage in which candidate answers are extracted from relevant documents and assigned a confidence score – i.e., the extractor confidence that the candidate answer is correct.
- Answers Generation – the stage in which candidate answers are combined based on notions of similarity and overlap, and then scored according to overall correctness confidence. The final ordered answer set is presented to the user.

There are different question types such as acronym, counterpart, definition, famous, stand for, synonym, why, name-a, name-of, where, when, who, what/which, how, yes/no and true/false [3]. Where, when, which, yes/no, true/false, and name of are kinds of factoid questions. As an example, ―whatis research?" is a definition question whereas ―where is Mt. Ras Dashen located?" is a where question which seeks location. Some questions are closed-domain (where the questions raised are in a specific domain such as in medicine) and open-domain which are questions almost about everything [3].

Researchers have followed many directions to answer factoid and non-factoid questions in open and closed domains question answering including: question parsing and classification, extracting answers from Web documents [5], statistical approaches to answer extraction and selection [6], semantic analysis, reasoning, and inference [7].

1.2 Statement of the Problem

There is huge amount of electronic Amharic documents. The urge to get as concise information as possible for natural language questions using computers leads to Question answering (QA). In contrast with traditional document retrieval systems/search engines, which return ranked lists of potentially relevant documents that users must then manually browse through, question answering systems attempt to directly provide users one or more concise answers in the form of sentences or phrases to natural language questions [1].

The field of question answering is substantially narrower and focuses on a few specific question types. Research on Amharic Question Answering focuses on factoid questions, whose answers are typically named entities such as dates, locations, proper nouns, other short noun phrases, or short sentences [3]. The following are a few examples of these so-called ―factoid‖ questions: "**የኢትዮጵያ የወንዶች እግር ኳስ ቡድን ለመጨረሻ ጊዜ በአፍሪካ ዋንጫ የተሳተፈው መቼ ነው? "** (―When was the last African cup of nations that the Ethiopian male football team participated?"), "**የዶ/ር ሐዲስ ዓለማየሁ የትውልድ ቦታ የት ነው?**" (―Where is the birth place of Dr. Hadis Alemayehu?") , "**የፍቅር እስከ መቃብር ደራሲ ማን ነው?**" (―Who is the author of Fikir Eskemekabir?"), "**በኢትዮጵያ የመንግስት ዩኒቨርሲቲዎች ብዛት ስንት ነው?**" (―How many public Universities are found in Ethiopia?"), whose answers are time, place, person name, and quantity, respectively.

But there are other types of questions that are non-factoid like: ―list" questions such as "**ሉሬት ፀጋዬ ገ/መድኅን የፃፉትውን ድርሰቶች ዘርዝር?**" (―List the literary works that Louret Tsegaye Gebremedihin wrote?"), which are similar to factoid questions but have multiple answers; biographical questions such as "**አርቲስት ጥላሁን ገሰሰ ማነው?**" (―Who is artist Tilahun Gessese?") which require a system to gather relevant encyclopedic facts about a person from multiple documents; and other complex questions (definitional questions, why questions,…) that require reasoning and fusion of multiple ―information pieces" from different sources. Amharic non-factoid questions are questions with complex answers; generally require definitions, descriptions, reasoning or procedural explanations. Specifically Amharic definitional, description and biographical questions can't be answered by Amharic factoid question answering because their answers are not entity names.

1.3 Objectives

1.3.1 General Objective

The general objective of this study is to investigate features of definitional, description and biographical questions and their answers and design an Amharic QA system to analyze and answer such Amharic questions.

1.3.2 Specific Objectives

The specific objectives are:

- Review related works in Amharic and other languages.
- Extract specific features of Amharic definitional, description and biographical questions and answers by investigating Amharic language issues in relation to them.
- Identify criteria along which questions taxonomies should be formed.
- Formulate algorithm for question type categorization.
- Review and select IR and NLP techniques that enhance the answer extraction process.
- Design theoretical models of the Amharic QA system that answer definitional, description and biographical Amharic questions.
- Develop prototype for the Amharic QA system.
- Prepare Amharic definition, description, and biography questions and documents that will be used for evaluation.
- Evaluate the Amharic definitional, description, and biographical QA system.

1.4 Methods

In order to achieve the objective of the study, the following methods will be employed.

Literature Review

Literature review has a vital role for identifying problems, finding gaps, identifying methodologies, etc. In addition, in order to understand the state-of-the-art in question answering, books, articles, journals and other publications will be reviewed.

Data Collection

In order to design an efficient Amharic QA system for definitional, description and biographical questions, analyzing the characteristics of these question types, their respective answers and the data source used for answering the questions is crucial.

Thus, for the purpose of training and testing the question classifier questions will be prepared from different Amharic documents. In addition, documents will be collected from different Amharic newspapers on the web and other official websites to evaluate the document retrieval and answer extraction.

Tools

In order to successfully achieve the research objectives, a number of tools are needed. Java programming language will be used for the development of the prototype. Java is selected for its suitability in developing applications and because it supports Unicode processing. Moreover it is an open source language.

A Lucene searching and indexing API will be used by the document retrieval module of our system. In addition, the stemmer in [3, 32] will be used for stemming and the HornMopho [50] for morphological analysis. The summarizer of [39] will also be used by the biography answer extraction component. Moreover, for the Named Entity Recognizer (NER), as there is no such tool easily available to integrate with our system, the gazetteer based NER developed in [3] will be used.

Testing

Performance evaluation will be conducted manually by comparing the system‘s answers with their corresponding manual answers of the questions prepared for testing. That is, the evaluation metrics like precision, recall, and F-score will be used for the evaluation.

1.5 Scope and Limitations

Current QA tasks are classified as factoid QA and non-factoid QA and also could be open-domain or closed domain. Furthermore, there are many classifications among non-factoid questions. This research will try to answer only Amharic definitional “**…ምን ማለት ነው**?” (―What is …?”)/ “**…ትርጉሙ ምንድን ነው**?” (―What is the meaning of …?”), description

"…ጥቅሙ ምንድን ነው?" (–What is the use of …?"), and biographical "… ማነው?/ማናት?/ ማናቸው?" (Who is …?) type questions. Moreover, this thesis will use only Amharic textual documents from electronic newspapers and other official websites without any figure, table, image or pictorial representations.

1.6 Application of Results

The research is expected to improve the understanding of Amharic question answering for definitional, description, and biographical questions. These questions require searching from multiple documents, fusion of information and drawing conclusion. This research tries to address these issues, and as a result will give a chance for researchers to grasp relevant information. Moreover, if there is a probability of implementing the result of this research in the future, it can be used in different areas to get precise information for information requests formulated as questions in Amharic, in which it will alleviate the problem of getting precise information in different fields. It can be applied in academic areas like in elementary and high schools.

1.7 Thesis Organization

The rest of the thesis is organized as follows. State of the art on question answering, information retrieval techniques, and related issues are covered in Chapter 2. Also, this chapter focuses on defining terminologies and elaborating facts to avoid ambiguities in the coming chapters. Chapter 3 critically reviews related question answering works in Amharic and other languages. It presents the gaps in the reviewed researches. It also proposes a solution to bridge the gaps. Chapter 4 presents a detailed description of our proposed QA model along with its implementation. Chapter 5 presents the evaluation environment and the evaluation criteria, and the empirical results of the proposed system along with their interpretations. Finally, Chapter 6 concludes the thesis with the research findings, conclusions, and future works.

Chapter Two: Literature Review

This chapter discusses the generic ideas about question answering system components, which are question analysis, document retrieval, answer extraction, and answer selection. It also gives short review on other topics related to QA, such as information retrieval and its models, and text summarization.

2.1 Question Answering (QA)

Research in Question Answering began in the early 1960s and since then, Question Answering has been the subject of interest of a wider community, from the fields of information retrieval, applied linguistics, human-computer interaction and dialogue [10]. QA research had a significant boost when it became the subject of interest of the annual Text REtrieval Conferences (TREC), a series of workshops promoted by the US National Institute of Standards (NIST) with the aim of advancing the state-of-the-art in text retrieval [2].

One of the first generic Question Answering algorithms, presented in the 1970s, consisted of the following steps: first, taking the set of documents on which to perform QA and accumulating a database of semantic structures representing the meanings of the sentences forming such documents; then, a set of structures sharing several lexical concepts with the question was selected to form a list of candidate answers; finally, the question was matched against each candidate structure and the best matching structure was selected as the answer [2].

This very simple approach shows the important role of semantic processing that has characterized Question Answering from its beginning, exploiting information other than facts available in database systems, and distinguished it from information retrieval.

Early work in the field of QA concerned on very limited domains and consisted in retrieving information from small databases, such as records of sport events and also the type of questions that were addressed were limited. Currently works are aiming to go beyond simple QA; they attempt to address definitional questions, more complex than questions requiring factoids such as names and dates, using different data sources on closed and open domains. That is the range of possible questions is not constrained, hence a much heavier challenge is placed on systems, as it is impossible to pre-compile all of the possible semantic structures appearing in a text [2].

From a very general perspective QA can be defined as an automatic process capable of understanding questions formulated in a natural language such as Amharic and responding exactly with the requested information.

However, this apparently simple definition turns out to be very complex when we analyze in detail the characteristics and functionality an "ideal" QA system should have. This system should be able to determine the information need expressed in a question, locate the required information, extract it, and then generate an answer and present it according to the requirements expressed in the question. Moreover, this ideal system should be able to interpret questions and process documents that are written in unrestricted-domain natural languages, which would allow for a comfortable and appropriate interaction by users [1].

2.1.1 QA General Architecture

QA systems cover a wide scope of different techniques, such as question type ontology, databases of external knowledge, heuristics for extracting answers of certain types, generation of answers, answer justifications, inference rules, feedback loops and machine learning. Hence, it is impossible to capture all variations within a single architecture. Since most systems have a number of features in common, this allows to give a general architecture of a question answering system.

According to [10], the main components of such a general architecture and the ways in which they interact are shown in Figure 2.1. The system has four components: question analysis, document retrieval, answer extraction, and answer selection.

This approach is often described as a QA pipeline, in which natural language questions flow into the first module, which is responsible for question analysis, and answers flow out of the last module, in which the answer is extracted and packaged into a response for the user. Modules are chained such that the output of an upstream module is connected directly to the input of the next adjacent downstream module.

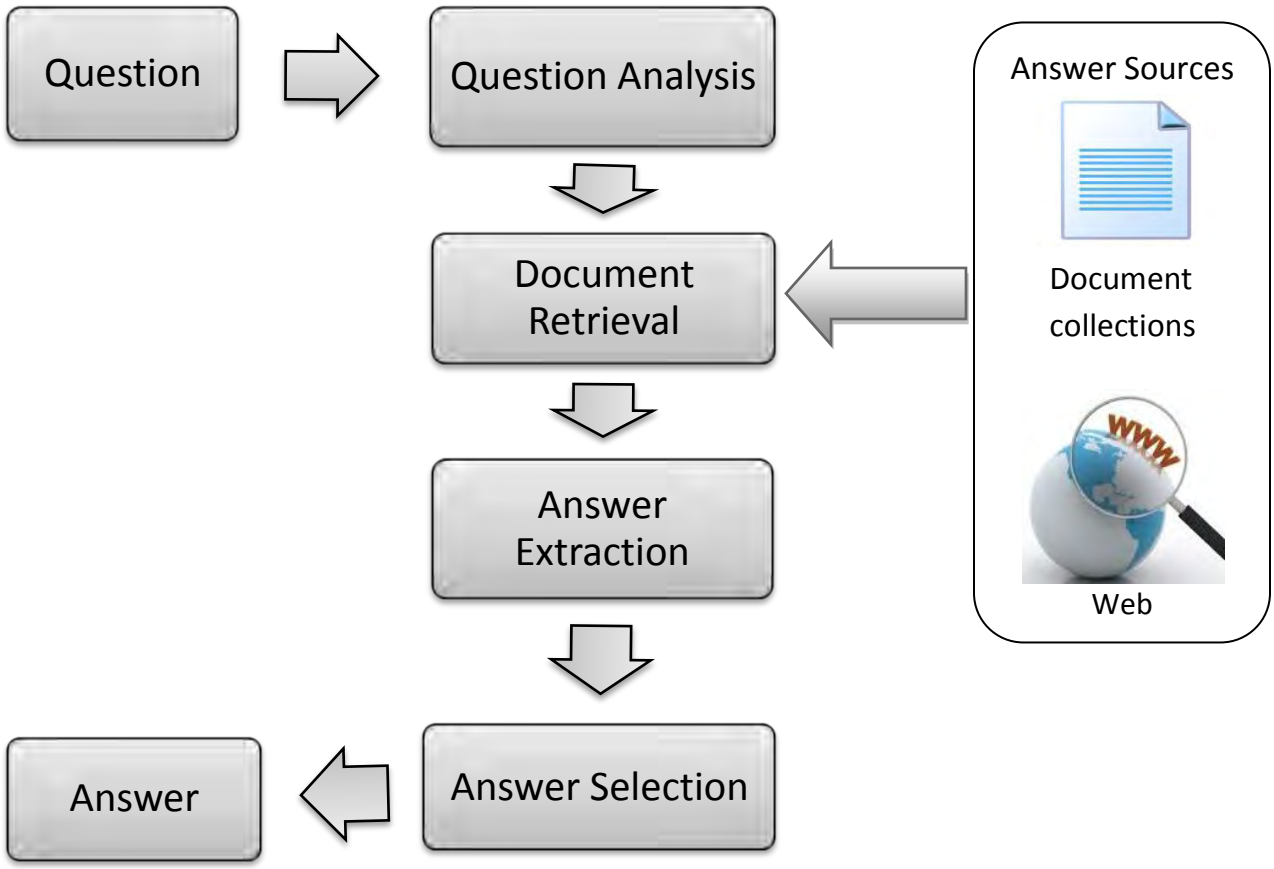

***Figure 2.1**: The General Approach to QA*

The first step is to analyze the question itself which is asked in a natural language by a user. The question analysis component may include at least a morphological analyzer or analysis of the question. The question is also classified to determine what it is asking for, i.e., whether it is asking for a date, a location, the name of a person, definition, description, etc. Depending on the morphological analysis and the class of the question, a retrieval query is formulated which is passed to the document retrieval component. Some of this information, such as the question class and a syntactic analysis of the question, are also sent to the answer extraction component.

The document retrieval component is generally a standard document retrieval system which identifies a subset of documents that contain terms of a given query from the total document collection. The retrieval component returns a set or ranked list of documents which are deemed

most likely to contain an answer to the question by comprising those documents. Then they are further analyzed by the answer extraction component.

The answer extraction component takes as input documents that are likely to contain an answer to the original question, together with a specification of what types of phrases should count as correct answers. This specification is generated by the question analysis component. The answer extraction component extracts a number of candidate answers which are sent to the answer selection component.

The answer selection component selects the phrase that is most likely to be a correct answer from a number of phrases of the appropriate type, as specified by the question analysis component. The candidate answers which are extracted from the documents are ranked in terms of probable correctness and are returned to the user [10].

2.1.2 Question Analysis

Analyzing the natural language question provided as input to the system is the first step towards finding the answer. The main function of the question analysis component is to understand the question. Thus the question is analyzed in a number of ways. First, a morphosyntactic analysis of the words in the question is carried out. This assigns to each word in the question a part-of-speech tag, indicating whether a word is a verb, singular noun, plural noun, etc. After having assigned part-of speech tags to words, it is useful information to judge the kind of information the question is asking for. In order to understand what the question asks for, identifying the semantic type of the question is an important step for extracting the actual answer, i.e., Question Classification [10].

Question Classification: is the process of putting the questions into several semantic categories. The set of possible classes is predefined, and ranges from few basic sets only depending on looking at the key question word, such as "… **ማነው?/ማናት?/ማናቸው?**" (who) a person's biography; "… **ምን ማለት ነው?**" (what) a meaning; "…**ጥቅሙ ምንድን ነው?**" (what) a description. For instance, if the system understands that the question "**ዶ/ር ሀዲስ ዓለማየሁ ማነው?** (Who is Dr. Hadis Alemayehu)" expects a person's profile/biography as answer, the search space of plausible answers will be significantly reduced.

The accuracy of question classification is very important to the overall performance of the Question Answering system. Thus, most systems resort to more detailed analysis of the question which determines additional constraints on the answer entity by, for example:

- identifying keywords in the question which will be used in matching candidate answer-bearing sentences or
- identifying relations, syntactic or semantic, that ought to hold between a candidate answer entity and other entities or events mentioned in the question.

Various systems have built hierarchies of question types based on the types of answer sought, and attempt to place the input question into the appropriate category in the hierarchy. Most question answering systems use a coarse grained category definition. Table 2.1 shows a number of question classes, it bears the question classes of many current Question Answering systems [1].

***Table 2.1**: List of question types*

የግለሰብ ስም (PERSON)	**የቦታ ስም** (PLACE)	**ቀን** (DATE)
ብዛት (NUMBER)	**ትርጉም** (DEFINITION)	**ድርጅት** (ORGANIZATION)
ማብራሪያ (DESCRIPTION)	**አጎጽሮተ-ጽሑፍ** (ABBREVIATION)	**መጠሪያ/ስም** (KNOWNFOR)
ርዝመት (LENGTH)	**የገንዘብ መጠን** (MONEY)	**ምክንያት** (REASON)
ዘዴ/ሂደት (METHOD/WAY)	**ጥቅም/ፋይዳ** (PURPOSE)	**ሌላ ዓይነት ጥያቄ** (OTHER)

Classification of questions can be implemented in a variety of ways. The simplest method is to use a set of rules that map patterns of questions into question types. The patterns are expressed by means of regular expressions on the surface form. The identification of the answer type is usually performed by analyzing the interrogative terms of the question (wh-terms). For example, given the question "**ጥናትና ምርምር ምን ማለት ነው**?" (What is research?) the term "**ምን ማለት**" (what) indicates that the question is looking for a definition. This approach may give

acceptable results with little effort for coarse-grained question taxonomy. Furthermore, the human-produced classification rules are easy to read by humans. However, the development of such rules can be very time-consuming when the question taxonomy is fine-grained or when very high accuracy is required. Thus a change in the domain of application will typically require a new set of classification rules that need to be built almost from scratch [1].

An alternative approach to question classification is the use of machine-learning techniques. With this approach, question classification is treated as a —standard" classification task that can be tackled with statistical packages of classification. Provided that there is a corpus of questions available together with annotations indicating the correct class of each question, this approach is fairly straightforward once a set of features has been identified.

Statistical approaches employ various mathematical techniques and often use large text corpora to develop approximate generalized models of linguistic phenomena based on actual examples of these phenomena provided by the text corpora without adding significant linguistic or world knowledge [27]. Methods for statistical NLP mainly come from machine learning, which is a scientific discipline concerned with learning from data. That is, to extract information, discover patterns, predict missing information based on observed information, or more generally construct probabilistic models of the data. Machine learning techniques can be divided into two types: supervised and unsupervised [1].

Supervised learning is mainly concerned with predicting missing information based on observed information. It employs statistical methods to construct a prediction rule from labeled training data. Näive Bayes, decision tree, neural network, and support vector machines (SVMs) are examples of supervised learning algorithms. The goal of unsupervised learning is to group data into clusters.

In supervised learning, the classifier, or its scoring function, is learned from a set of labeled examples, referred to as training data. Its performance is evaluated on a separate set of labeled data called test data. The idea of finding an optimal separating hyper plane with largest margin leads to a popular linear classification method called SVM [28, 29, 30].

SVM is basically a binary classifier. It takes a set of input data and predicts, for each given input, which of two possible classes forms the output. Given a set of training examples, each marked as

belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on [31, 32]. SVM is proved to give better performance for text classification purpose compared to other supervised classification algorithms [28, 32, 33, 34].

Query Generation

Question-analysis also extracts the information in the question that allows the generation of queries that could be used by an IR system for the extraction of the answer-bearing text from the entire document collection. These queries are commonly obtained using keyword selection and answer-pattern generation processes [1].

- **Keyword selection** focuses on selecting those question terms whose appearance in a text indicates the existence of a candidate answer in the surrounding text. For the question "**የኢትዮጵያ ኦርቶዶክስ ተዋህዶ ቤ/ን ፓትርያርክ የነበሩት ብፁዕ ወቅዱስ አቡነ ተክለ ሃይማኖት ማናቸው?**" (Who was his Holiness Abune Tekle Haimanot the Patriarch of Ethiopian Orthodox Tewahido Church?), the keyword group would be formed by the terms, "**የኢትዮጵያ ኦርቶዶክስ ተዋህዶ ቤተክርስቲያን**" (Ethiopian Orthodox Tewahido Church), "**ፓትርያርክ**" (Patriarch) and "**አቡነ ተክለሃይማኖት**" (Abune Tekle Haimanot). This set of keywords may then be expanded using synonyms and/or morphological variants.
- **Answer pattern generation** attempts to generate queries that constitute one or several combinations of question terms in such a way that each resulting query expresses a form in which the answer might be found. The following possible queries could be derived from the previous example question: The birth place of ―His Holiness Abune Tekle Haimanot is X in Y", ―Abune Tekle Haimanot became Patriarch on X",…, where X and Y refer to the expected answer. The generated query expressions do not need to be linguistically perfect, since they are fed to an IR system that usually treats the expressions as bags of words.

2.1.3 Document Retrieval

QA systems have document retrieval component to narrow the search space for an answer. Its main purpose is to select an initial set of candidate answer-bearing documents from a large text collection prior to sending them to a downstream answer extraction module.

In an effort to pinpoint relevant information more accurately, the documents are split into several passages, and the passages are treated as documents. Thus, many QA systems also have a passage retrieval stage, interposed between the document retrieval and answer extraction components, which can be thought of as a second, smaller scale IR module. Using a passage-based retrieval approach instead of a full-document retrieval approach has the additional advantage that it returns short text excerpts instead of full document which are easier to process by later components of the question answering system.

Document retrieval has a long tradition and many frameworks have been developed over the years, resulting in sophisticated ways to compute the similarity between a document and a query. Depending on the retrieval engine that is actually used, the retrieval component returns either an unordered set of documents that are likely to contain an answer, or a ranked list of documents, where the documents are ranked with respect to their likelihood of containing an answer.

Document retrieval is just one of the components of the whole question answering process; its effectiveness is critical to the overall performance of a question answering system. If the document retrieval component fails to return any document that contains an answer, even optimally functioning answer extraction and answer selection components will inevitably fail to return a correct answer to the user [10].

2.1.4 Answer Extraction

Answer extraction is the act of extracting text strings constituting the exact answer or answers to a question from candidate answer-bearing document(s) retrieved by the document retrieval module. The answer extraction module performs detailed analysis and pin-points the answer to the question as specified by the question analysis component. Usually the answer extraction module produces a list of answer candidates and ranks them according to some scoring functions [11].

Answer extraction can also be viewed as a classification problem in which answer correctness is evaluated based on various features derived from both the question and the candidate answer. QA systems like Javelin use support vector machine, k-nearest neighbor, and decision tree classifiers to evaluate the correctness of individual answers [51]. Thus under this approach candidate answers are identified and then their correctness assessed by different classifiers for different answer types. A maximum-entropy approach is another approach that has been used by different QA systems [52]. The maximum-entropy approach based model uses sentence features, entity features, definition features, and linguistic features in order to capture various aspects of the question-answer relationship. Another approach to answer extraction is to apply simple patterns based on surface forms or part-of speech tags in order to build an off-line database for specific question types – e.g., who-is type questions. Moreover, many QA systems exploit redundancy of candidate answers by using answer frequency to boost the confidence of correct answers [4].

2.1.5 Answer Selection

After question classification, document retrieval, and answer extraction, question answering systems are faced with a set of text fragments which are possible correct answers. These candidates usually originate from different passages and are often extracted using different strategies. Moreover, these textual fragments may not always constitute full answers.

The set of answer candidates obtained through answer extraction could include [4]:

- *Complete answer* – text snippet which fully satisfies the information need of the user.
- *Overly specific answer* – text snippet with a lower granularity than that of the expected answer, partially satisfying or not satisfying the user‘s information need.
- *Overly generic answer* – text snippet with a higher granularity than that of the expected answer, partially satisfying or not satisfying the user‘s information need.
- *Partial answer* – text snippet containing part of the information required to answer the question. Partial answers from multiple documents or passages need to be combined in order to generate a complete answer.

Answering definitional, biographical, description, how, why and other complex questions require to put together partial answers from different documents, in which this task is handled by the

answer generator. So answer generation is the task of taking the output of answer extraction (i.e., candidate answer components) and produce correct and complete answers with corresponding confidence scores. This task involves combining evidence from several answer candidate components in order to generate meaningful correct answers with high confidence scores [4].

Similar to the other components, there are a variety of ways to select or rank candidate answers. After a text unit containing an expected answer type has been found, other constraints are applied to the text unit. That is once an expression of the correct answer type is found in a candidate answer-bearing paragraph, an answer window around the candidate would be established and various quantitative features such as word overlap between the question and the answer window could be used in a weighted numerical heuristic to compute an overall score for the window. Thus, for every candidate answer-bearing paragraph which contains an expression of the correct answer type, a score is derived for the answer-window containing the answer candidate and these scores are used to compute an overall ranking for all answer candidates [10].

In addition to word-based comparison approaches, the frequency of a candidate answer can also be considered as a criterion for answer selection. The frequency of a candidate answer is the number of occurrences it was linked to the question. Using frequencies to select an answer is also known as redundancy-based answer selection. Counting these frequencies can be restricted to the set of documents that were considered in the answer extraction component, but it can also be extended to a larger set. Some question answering systems use the whole document collection to count how often a candidate answer co-occurs with terms from the question. Other systems even go beyond the actual document collection and use the World Wide Web to get these frequencies [10].

2.2 Information Retrieval

Information retrieval (IR) is finding documents of an unstructured nature (usually text) that satisfy the information need from within large collections (usually stored on computers) [12].

Question Answering Systems have evolved from the field of Information Retrieval. Information Retrieval traditionally takes as input a set of keywords that constitutes a query. The query is then processed by various algorithms. Usually the query is broken up by separating all the individual keywords which are then used to search for relevant documents. During the search, the keywords will be matched up against an index that references the different documents. Many times a keyword query will not contain sufficient information to produce quality results. One reason can be due to the lack of keywords or the lack of descriptive keywords. This is a very likely scenario when clients type in the queries manually. These types of problems can be solved to a certain degree by applying query rewriting, or query expansion prior to searching. There are several different algorithms that are involved in this process such as finding keyword synonyms, and possibly apply stemming algorithms [13]. Stemming is used to obtain a word‘s morphological root, thereby reducing the granularity of words that need to be indexed. It is usually an integral part in any modern IR system.

In general IR systems take two main inputs, the user needs and the information items. It typically returns one main output consisting of a ranked list of information items. The three main processes in an IR system are: extraction of item content features and descriptors into a logic representation of items (indexing); handling user‘s information needs into an abstract representation (query processing) and matching both representations (searching and ranking) [14].

- **Indexing**: Not all the pieces of an information item are equally significant for representing its meaning. In written language, for example, some words carry more meaning than others. Therefore, it is usually considered worthwhile to pre-process the information items to select the elements to be used as index objects. Indices are data structures constructed to speed up search. It is worthwhile building and maintaining an index when the item collection is large and semi-static. The most common indexing structure for text retrieval is the inverted file. This structure is composed of two elements: the vocabulary and the term occurrences. The vocabulary is the set of all words in the text. For each word in the vocabulary a list of all the

text positions where the word appears is stored. The set of all those lists is called occurrences.

- **Query processing**: The user needs, the query, is parsed and compiled into an internal form. In the case of textual retrieval, query terms are generally pre-processed by the same algorithms used to select the index objects. Additional query processing (e.g., query expansion) requires the use of external resources such as thesauri or taxonomies.
- **Searching**: user queries are matched against information items. As a result of this operation, a set of potential information items is returned in response to user needs. The way this is achieved may vary considerably depending on the format of information (text, audio, video, etc.), but in all cases, some form of simplification is done in the information model to make it tractable. For instance, text retrieval commonly builds on the assumption that the matching between information items (the documents) and user information needs (the query string) can be based on a set of index terms. This obviously involves a loss of semantic information when text is replaced by a set of words. A similar situation occurs in multimedia retrieval where matching is performed based on numeric signal features.
- **Ranking**: The set of information items returned by the matching step generally constitutes an inexact, by nature approximate answer to the information need. Not all the items contain relevant information to the user. The ranking step aims to predict how relevant the items are comparatively to each other, thus returning them by decreasing the order of estimated relevance. Thus, in a way, ranking algorithms can be considered the core of IR systems, as they are keys to determine the performance of the system.

Information Retrieval Models

The ranking algorithm is one of the main characteristic components of an IR system. A ranking algorithm operates according to basic premises regarding the notion of document relevance. Distinct sets or premises yield different IR models. This section tries to cover three of the most important classic text IR models, namely: Boolean, Vector, and Probabilistic.

In the Boolean model, documents and queries are represented as a set of index terms. In the Vector space model, documents and queries are represented as vectors in a t-dimensional space. In the basic probabilistic model, documents and queries representations are based on probability

theory. Following the definition in [15], [14] states an IR model is a quadruple [D, Q, F, sim], where:

- D is a set of (logical representations of) documents.
- Q is a set of (logical representations of) queries.
- F is a framework for modeling documents, queries, and their relationships.
- sim: $Q \times D \rightarrow U$ is a ranking function that defines an association between queries and documents, where U is a totally ordered set (commonly [0, 1], or P, or a subset thereof). This ranking and the total order in U define an order in the set of documents, for a fixed query.

A. Boolean Model

The Boolean Model is a simple retrieval model based on set theory and Boolean algebra. Documents are represented by index terms extracted from documents, and queries are Boolean expressions on terms. Following the previous notation, here [14]:

- D: the elements of D are represented as sets of index terms occurring in each document. Terms are treated as logic propositions, denoting whether the term is either present (1) or absent (0) in the document. Documents can thus be seen as the conjunction of their terms.
- Q: queries are represented as a Boolean expression composed by index terms and logic operators (AND$\wedge$, OR$\vee$, and NOT$\neg$) which can be normalized to a disjunction of conjunctive vectors.
- F is a Boolean algebra over sets of terms and sets of documents.
- sim is defined by considering that a document is predicted to be relevant to a query if its index terms satisfy the query expression.

The Boolean model suffers from two major drawbacks. First its retrieval strategy is based on a binary criterion (i.e., a document is predicted to be either relevant or non relevant), the index terms can only be given Boolean weights i.e., $w_i \in \{0, 1\}$. This means that with too restrictive expressions no documents will qualify. On the other hand a very general expression will result in too many documents being returned. Therefore, it does not provide a proper basis for ranking the retrieved results, which may likely result in low precision levels when the retrieval space is too big. Second, it is not always easy for most users to translate an information need into a Boolean expression with logic operators [14].

B. Vector Space Model

The vector space model (VSM) recognizes that the use of binary weights is too limiting and proposes a framework in which partial matching is possible. This is accomplished by assigning non-binary weights to index terms in queries and documents. These term weights are ultimately used to compute the degree of similarity between each document stored in the system and the user query. By sorting the retrieved documents in decreasing order of this degree of similarity, the VSM takes into consideration documents which match the query terms only partially. The main resulting effect is that the ranked document answer set is considerably more precise (in the sense that it better matches the user information need) than the answer set retrieved by a Boolean model [14].

Following the previous notation:

- D: documents are represented by a vector of words or index terms occurring in the document. Each term in the document – or, for that matter, each pair (t_i, d_j) – has a positive, non-binary associated weight $w_{i,j}$.
- Q: queries are represented as a vector of words or index terms occurring in the query. Each term in the query– or, for that matter, each pair (t_i, q) – has a positive, non-binary associated weight w_i .
- F is an algebraic model over vectors in a t-dimensional space.
- sim estimates the degree of similarity of a document d_j to a query q as the correlation between the vectors d_j and q. This correlation can be quantified, for instance, by the cosine of the angle between the two vectors:

$$sim(\vec{q}, \vec{dj}) = \cos(\vec{q}, \vec{dj}) = \frac{\vec{q} \cdot \vec{dj}}{|\vec{q}| * |\vec{dj}|}$$

  where t_i is index term for document d_j.

Since $w_{i,j} > 0$ and $w_i,q > 0$, sim(q, d_j) varies from 0 to 1. Thus, instead of attempting to predict whether a document is relevant or not, the VSM ranks the documents according to their degree of similarity to the query. A document might be retrieved even if it matches the query only partially.

C. Probabilistic Model

The probabilistic model aims to capture the IR problem in a probabilistic framework. The fundamental idea is as follows. Given a query q and a collection of documents D, a subset R of D is assumed to exist which contains exactly the relevant documents to q (the ideal answer set). The probabilistic retrieval model then ranks documents in decreasing order of probability of belonging to this set, which is noted as P (R |q, d_j), where d_j is a document in D [14].

Following the previous notation:

- D: documents are represented as a vector of words or index terms occurring in a document. Each term in the document, that is, each pair (t_i, d_j), has a binary associated weight 1 or 0, denoting the presence or absence of the term in the document.
- Q: queries are represented by a vector of words or index terms that occur in the query. Each term in the query, that is, each pair (t_i, q) has a binary weight 1 or 0, denoting the presence or absence of the term in the query.
- F is a probabilistic model that ranks documents in order of probability of relevance to the query.
- sim measures the degree of similarity of a document d_j to a query q_i as the probability of d_j to be part of the subset R of relevant documents for q. This is measured in the probabilistic model as the odds of relevance, as given by: $sim(d_j, q) = \frac{P(R|d_j)}{P(\neg R|d_j)}$ where $\neg R$ denotes the set of non relevant documents, $P(R|d_j)$ is the probability of d_j being relevant to the query q, and $P(\neg R|d_j)$ is the probability of d_j being non relevant to q.

2.3 Text Summarization

Text summarization is the process of distilling the most important information from a text to produce an abridged version for a particular task [53]. Important kinds of summaries that are outlines of any document, abstracts of a scientific article, headlines of a news article, snippets summarizing a web page on a search engine results page, action items or other summaries of a (spoken) business meeting, summaries of email threads, compressed sentences for producing simplified or compressed text and answers to complex questions, are constructed by summarizing multiple documents.

These kinds of summarization goals are often characterized by their position on two dimensions, which are single document versus multiple document summarization and generic summarization versus query-focused summarization [54, 56].

In single document summarization, a single document is given and produces a summary. Single document summarization is thus used in situations like producing a headline or an outline, where the final goal is to characterize the content of a single document. Whereas in multiple document summarization, the input is a group of documents, and the goal is to produce a condensation of the content of the entire group. It is used when to summarize a series of news stories on the same event or web content on the same topic [54].

A generic summary is one which does not consider a particular user or a particular information need; the summary simply gives the important information in the document(s). By contrast, in query-focused summarization, also called focused summarization, topic-based summarization and user-focused summarization, the summary is produced in response to a user query. We can think of query-focused summarization as a kind of longer, non-factoid answer to a user question [56].

One crucial architectural dimension for text summarizers is whether they are producing an abstract or an extract. The simplest kind of summary, an extract, is formed by selecting (extracting) phrases or sentences from the document to be summarized and pasting them together. In contrast, an abstract uses different words to describe the contents of the document [55].

Text summarization systems and natural language generation systems as well, are generally described by their solutions to the following three problems [1]:

- Content Selection: What information to select from the document(s) we are summarizing. Content selection thus mainly consists of choosing which sentences or clauses to extract into the summary.
- Information Ordering: How to order and structure the extracted units.
- Sentence Realization: What kind of clean up to perform on the extracted units so they are fluent in their new context.

Most information needs are formulated by questions. Also most interesting questions are non factoid questions. User needs require longer, more informative answers than a single phrase. For example, a DEFINITION question might be answered by a short phrase like "*Autism is* a developmental disorder", a user might want more information, as in the following definition of *water spinach*:

> *"Water spinach* (ipomoea aquatica) is a semi-aquatic leafy green plant characterized by long hollow stems and spear-shaped or heart-shaped leaves which is widely grown throughout Asia as a leaf vegetable. The leaves and stems are often eaten stir-fried as greens with salt or salty sauces, or in soups." [10].

Similarly, a BIOGRAPHY question like *who was artist Tilahun Gessese?* instead of a factoid answer like *Tilahun Gessese was a singer*, might produce a brief biography:

> "Tilahun Gessesse (September 29, 1940-April 19, 2009) was an Ethiopian singer regarded as one of the most popular of his country's "Golden Age" in the 1960s. During the 1960s he became famous throughout the country, nicknamed "The Voice". He raised money for aid during the famines of the 1970s and 1980s and earned the affection of the nation, being awarded a honorary doctorate degree by the University of Addis Ababa and also winning a lifetime achievement award from the Ethiopian Fine Art and Mass Media Prize Trust. In his later years he suffered from diabetes. He died on 19 April 2009 in Addis Ababa shortly after returning from America. Tilahun was honored with a state funeral attended by tens of thousands of his fellow citizens." [16].

Questions can be even more complex than the above questions, where a factoid answer might be found in a single phrase in a single document or web page. These kinds of complex questions are likely to require much longer answers which are synthesized from many documents or pages.

For this reason, summarization techniques are often used to build answers to these kinds of complex questions. When a document is summarized for the purpose of answering some user query or information need, it is called query-focused summarization [56]. The terms topic-based summarization and user-focused summarization are also used. A query-focused summary is thus really a kind of longer, non-factoid answer to a user question or information need.

One kind of query-focused summary is a snippet, the kind that web search engines like Google return to the user to describe each retrieved document. Snippets are query focused summaries of a single document. But for complex queries summarizing and aggregating information from multiple documents is needed.

Any extracted sentence must contain at least one word overlapping with the query. Or just add the cosine distance from the query as one of the relevance features in sentence extraction. Such a method of query-focused summarization is characterized as a bottom-up, domain-independent method. An alternative way to do query-focused summarization is to make additional use of top-down or information-extraction techniques, building specific content selection algorithms for different types of complex questions.

In order to build extractive answers for definition questions, extract sentences with the genus information, the species information, and other generally informative sentences. Similarly, a good biography of a person contains information such as the person‘s birth/death, fame factor, education, nationality, and so on. Thus extracting sentences that have such kinds of information would enable to build the correct answer [10].

In this thesis we have used the summarizer of [39]. The author implemented an Amharic summarization system based on LSA (Latent Semantic Analysis) and graph-based algorithms which were tested on news texts. The author proposed two new algorithms that are called TopicLSA and LSAGraph.

2.4 Lucene

Apache Lucene[1] is a high-performance, full-featured text search engine library written entirely in Java. The library is released by the Apache Software Foundation under the Apache Software License. Lucene, being a search engine API, allows the creation of a vector-space model index which can then be searched with a large variety of query tools, including boolean, phrase, wildcard, and fuzzy match queries. Of particular use is Lucene‘s capability for searching hierarchical ―span phrase” queries, which are used to search for sentences with long range dependencies [24].

[1] http://apache.lucene.org/

2.4.1 Lucene's Indexing Process

Preprocessing operations (like character normalization, stop-word removal, short word expansion, and stemming or morphological analysis) are applied on the text before it is indexed in Lucene index. These pre-processing operations are called Analysis in Lucene terminology [25].

Indexing is a way of creating cross-reference lookup (index) in order to facilitate searching. Since Lucene's index lists the documents that contain a term, it falls into the family of indexes known as an inverted index. Inverse document frequency reflects how frequent a term is in the whole collection. The underlying principle is that a term that appears in a few documents gives more information than a term that appears in many documents. This means a term that appears in many documents has a weak discriminating power to select relevant documents over a document collection [26].

As stated in [3], IndexWriter, Directory, Analyzer, Document, and Field are the five basic classes of Lucene's indexer.

- IndexWriter: creates a new index and writes documents to an existing index.
- Directory: represents the location of a Lucene index. It is an abstract class that allows its subclasses to store the index. IndexWriter then uses one of the concrete Directory implementations, FSDirectory or RAMDirectory, and creates the index in a directory in the file system.
- Analyzer: is in charge of performing preprocessing of the data. Before text is indexed, it is passed through an Analyzer. The Analyzer class is language dependent as the preprocessing operations are language specific.
- Document: represents a collection of fields. It can be considered as a virtual document – a chunk of data, such as a web page, an email message, or a text file – that we want to make retrievable at a later time.
- Field: represents the document or metadata associated with that document. The metadata such as author, title, subject, date modified, and so on, are indexed and stored separately as fields of a document. Each field corresponds to a piece of data that is either queried against or retrieved from the index during search.

2.4.2 Searching Using Lucene

Searching is the process of looking for words in the index and finding the documents that contain those words. The search capabilities provided by Lucene allow the user to perform three different kinds of searches [14]:

- Exact search: the index must contain the exact searched term to retrieve an answer.
- Fuzzy search: the keywords stored in the index must be ―similar" to the searched term. The similarity is computed using the Levenshtein distance and considering an established prefix that represents the number of letters that must be equal at the beginning of both words.
- Spell search: the searched term may contain some mistakes; therefore Lucene provides suggestions of additional terms.

The core classes of the search API that facilitate searching are IndexSearcher, Term, Query, and Hits [25]. These classes are briefly discussed in [3].

- IndexSearcher is used to search from the index using different searching methods.
- A Term is the basic unit for searching. Similar to the Field object, it consists of a pair of string elements, the name of the field and the value of that field.
- Query is an abstract base class for queries. Searching for a specified word or phrase involves wrapping them in a term, adding the terms to a query object, and passing this query object to IndexSearcher's search method.
- The Hits class is a simple container of pointers to ranked search results; that is documents which match a given query.

Chapter Three: Related Work

This Chapter presents a review on Amharic factoid QA and non-factoid QA on English, Japanese, Arabic, and Chinese languages. It also incorporates summary of the chapter.

3.1 Question Answering for Amharic

The work in [3] investigated Amharic language specific issues extensively for document normalization and questions processing and developed ―TETEYEQ" Amharic QA for factoid questions which is designed based on the standard pipeline QA system by adding a number of techniques and approaches. The system is comprised of document pre-processing, question analysis, document retrieval, sentence/paragraph ranking, and answer selection components. In Amharic words are written using ―fidel". Some characters and punctuation marks have identical usage but different structural appearance. Moreover, numbers and dates can be written using Amharic numbers, Arabic numbers or alphanumeric. So document preprocessing was used to avoid such discrepancy within documents and questions.

The question analysis module produces a logical query representation, question focus (words or group of words that give hint about question entity), and expected answer types. The combination of predefined question particles/interrogative words and question focus were used by the question classification algorithm to identify the question type and the expected answer type. Sentence, paragraph, and file based document retrieval techniques were used to compare the generated queries with documents. Hence, a document with more number of terms matched with the query terms receives higher score and is passed to the sentence/paragraph ranking module that ranks paragraphs and re-ranks sentences according to the question types and the expected answer type. In addition, Gazetteer and pattern/rule based techniques were used to extract answer particles from the filtered documents.

Then the answer selection module extracts the best answer from the set of candidate answers from the ranked documents based on occurrence count of answer particles. The best answer particle is assumed to occur in more than one document. On the other hand the correct answer for a question might be sited in one document precisely and none in other documents. Thus in order to address such cases, another answer selection technique is used. That is considering the first answer particle from the ranked documents and inspect whether it has relevant answer or not.

The work in [32] presents "ጠይቀኝ (Ask Me)" web based factoid Amharic question answering system. The QA system contains web crawling, document preprocessing, question analysis, document retrieval, and answer extraction components. The web crawling component retrieves web pages and downloads the content. During this process pages that are not Amharic are filtered by the language identifier subcomponent of the crawler. The question analysis component identifies question type and generates query. The question classifier that identifies question types is implemented using machine learning approach. The performance is better than the classifier of [3], in which the classifier of [3] was implemented using rule based algorithm. The document preprocessing, document retrieval, and the answer extraction components are the same as that of the components in [3].

3.2 Non-Factoid Question Answering for English

The work in [17] describes a QA system for English known as DefScriber that answers definitional questions of the form "What is X?" using goal driven and data driven methods. The main stages in DefScriber operation are input, document retrieval, predicate identification, data-driven analysis, and definition generation.

The document retrieval module uses a fixed set of patterns to identify the term to be defined in the question, and then generates a set of search queries. These queries are sent to a web search engine until the specified number of documents is retrieved. Once documents are retrieved, goal-driven analysis is performed to identify predicates. The system examines documents for the instances of the three definitional predicates: Non-specific Definitional (NSD) (any type of information relevant in a detailed definition of a term), Genus (category to which the term belongs) and Species (describes properties other than or in addition to genus). Machine learning and rule (lexico-syntatic) based approaches were used to extract predicates.

The data-driven analysis uses techniques from summarization to cluster and order the entire set of NSD sentences based on properties of the data set as a whole. Then the definitional answer was generated by combining predicate information and data driven analysis or summarization result.

The work in [18] introduced a QA scheme that answers definitional questions of the form "what is X?" and "who is X?". The system first finds the target term (the concept for which information

is being sought). A simple pattern-based parser was used to extract the target term using regular expressions. If the natural language question did not fit into any of the patterns, the parser heuristically extracts the last sequence of capitalized words in the question as the target. Then nuggets relevant to the target term are extracted using database lookup, web dictionary lookup, and document lookup technique. Finally answers from the different sources are merged to produce the final output.

The target, the pattern type, nugget and source sentence are stored in a relational database. Then the database lookup technique answers definitional questions simply by looking for relevant nuggets in the database using the target terms as query. The dictionary lookup defines questions using Merriam Webster online dictionary. Keywords from the target terms and the target itself are used as the query to Lucene IR engine. Then filters and tokenizes the top one hundred documents into sentences and scored each based on their keyword overlap and inverse document frequency. The document lookup technique employs traditional document retrieval to extract relevant nuggets if no answers are found by the other two techniques. Finally, the answer merging component merges results from all the three sources. A simple heuristic was used to avoid redundancy, i.e., if two responses share more than sixty percent of their keywords, then one of them is randomly discarded.

The work in [19] introduces a QA system for English that goes beyond answering place, person/organization name, time, and quantity questions. The various statistical models of the QA system are trained using a corpus created by collecting question and answer pairs from Frequently Asked Questions (FAQ) on web pages. The QA system is composed of Question2Query, search engine, filter, and answer extraction modules.

The Question2Query Module transforms the posed question into phrase based query using a statistical chunker. The queries are sent to MSNSearch[2] and Google[3] search engines to retrieve relevant documents. Then the filter module tokenize and filters the selected documents by assessing the likelihood that a proposed answer is indeed an answer to the posed question. Then text fragments deemed to be the best answer to the question from the filtered documents are selected by the answer extraction module using n-gram co-occurrence and statistical translation

[2] http://search.msn.com
[3] http://www.google.com

based answer extraction algorithms. The n-gram co-occurrence based answer extractor assesses the likelihood that a proposed answer is a well-formed answer by calculating the n-gram overlap score for each proposed answer. The statistical translation based answer extractor generates an answer using a noisy channel model. That is the answer generation model proposes an answer A according to an answer generation probability distribution and an answer A is further transformed into question Q by an answer/question translation model according to a question-given-answer conditional probability distribution.

3.3 Non-Factoid Question Answering for Arabic

The work in [20] presents a definitional QA system for the Arabic language called DefArabicQA that identifies and extracts the answers from Web resources (Arabic version of Google and Wikipedia) using rule-based approach. It is composed of question analysis, passage retrieval, definition extraction, and ranking candidate definitions.

The topic of the question and the expected answer type were deduced using lexical question patterns and interrogative pronoun of the question, respectively. Top-n snippets retrieved by the Web search engine for the specific query were inspected and those snippets containing the question topic are kept. Then the definition extractor identifies and extracts candidate definitions from the collection of snippets using manually constructed lexical patterns. After filtering the extracted candidate definitions, the extractor ranks them by using global score. Finally the top five ranked candidate definitions are presented to the user. The performance of DefArabicQA is assessed by Arabic native speakers and mean reciprocal rank is also used as evaluation metric.

3.4 Non-Factoid Question Answering for Japanese

The work in [21] describes a system for answering non factoid Japanese questions by using answer type based weighting passage retrieval methods. They classified the non-factoid questions as definition-oriented, reason-oriented, method-oriented, degree-oriented, change-oriented and detail-oriented questions and used a particular method for each category. The system comprises of prediction of type of answer, document retrieval, and answer extraction.

The system predicts the answer type of a question based on the interrogative phrase and extracts terms from the input question by using morphological analyzer. The score of every document is calculated and the top 300 documents with higher score or that are likely to contain the correct

answer are gathered during the retrieval process to be used by the answer extractor. Then the answer extractor chunks the retrieved documents into paragraphs and retrieves those that contain terms from the input question and a clue expression. As a result, the system outputs the retrieved paragraphs as the preferred answer.

The work in [22] introduced two approaches to answer non-factoid questions. The first one has a monolithic architecture that retrieves answer passages related to a question using lexical chain. The other one has a type-by-type architecture.

The first approach, the one that has a monolithic architecture, processes all types of questions including factoid questions. The system applied a set of manually constructed lexico-syntactic patterns for classifying the input question as definition, why, how, or other, and to filter retrieved documents. Then it extracts passages related to the topic of the question using the information about lexical chains related to the topic, and assigns a relevance score to each retrieved passage. Finally it calculates a unified score for each candidate answer by combining the score of appropriateness, the score of relevance, and the score of document retrieval. Then, it selects a set of final candidate answers by using clustering-based summarization technique.

The system that works based on type-by-type architecture determines the type of question using the lexico-syntatic pattern for classification and directs the question to the respective subsystem for the answer. The subsystem for definitional and other-type questions is almost the same as the system with the monolithic architecture. But the subsystems for how-type questions and why-type questions have different components from the system with the monolithic architecture. They used the information of relative position between a key sentence (a sentence that contains keywords/question target or clue predicate) and an answer passage rather than the information of lexical chain. They adopted an existing factoid-type question-answering system for factoid questions.

3.5 Non-Factoid Question Answering for Chinese

The work in [23] presents a QA system that is capable of mining textual definitions from large collections of Chinese documents. In order to automatically identify definition sentences from a large collection of documents, they utilized the existing definitions in the Web knowledgebases instead of hand-crafted rules or annotated corpus. The QA system consists of question

processing, document processing, Web knowledge acquisition, definition extraction, and an optional module for corpus information acquisition.

The question processing module extracts the question target/target term and the document processing module retrieves documents from the corpus, filters, and chunks the retrieved documents into sentences. Then extracts relevant sentences to the target and removes any redundancy based on the percentage value of shared content words between sentences. On the other hand, the Web knowledge acquisition module acquires the definitions of the target term from the Web knowledgebase (specific websites on the Web that give biography of a person, the profile of an organization or the definition of a generic term). The GDS (General Definition Set) is formed by combining each definition from all Web knowledgebases. The elements of GDS could overlap in some degree. So to avoid redundancy, a new set called EDS (Essential Definition Set) is formed by merging GDS elements into one concise definition. Hence, the definition extraction module extracts the definition from the candidate sentence set based on the knowledge which is obtained from the Web knowledgebase and provides the answer to the user.

Corpus information acquisition module was used to rank the candidate sentences if no definition is found from the Web knowledgebases. It forms the centroid of the candidate sentence set and the sentences that have high similarity with this centroid are extracted as the answers to the question.

3.6 Summary

―TETEYEQ" and ―LETEYEQ" Amharic QA systems answer place, person/organization name, quantity, and date questions, i.e., factoid type questions by extracting entity names for the questions from documents using rule based answer extraction approach. Amharic non-factoid questions are questions with complex answers that require definitions, descriptions, reasoning or procedural explanations. Specifically Amharic definitional, description and biographical questions can't be answered by the Amharic factoid question answering because their answers are not simply entity names.

On the other hand, the reviewed non-factoid English, Arabic, Japanese, and Chinese QA systems are language dependent, i.e., only answer their respective language non-factoid question types. Since issues related to Amharic language are different from these languages, in addition to that

the question and answer patterns of non-factoid Amharic questions is different from those languages, the QA systems can‘t answer Amharic definitional and description questions. Thus, in this research we aim to develop an Amharic QA system that can answer Amharic biographical, definitional, and description questions.

Chapter Four: System Design and Implementation

This Chapter gives the architectural design and implementation of the definition, biographical and description Amharic Question Answering system. The design and implementation process of the proposed QA system consists of four major phases which are document preprocessing, question analysis, document analysis, and answer extraction. Document preprocessing includes character normalization, short word expansion, stop word removal, stemming, morphological analysis, and indexing. Question analysis includes question classification, query generation, and query expansion. Document analysis contains document retrieval and document filtering.

The fourth phase in the design and implementation process is answer extraction. In this thesis we propose a type-by-type approach for answer extraction. The definition-description answer extraction component extracts sentences/snippets using manually crafted definition and description answer extraction patterns, computes the score of the extracted sentences accordingly, selects non-redundant top ranked sentences using answer selection algorithm, and orders them to keep their coherence. On the other hand, the biography answer extraction component generates an answer using a summarizer. Detailed explanation of the design and implementation of these modules and the tools and methods, and the algorithms that are used by the different components of the proposed QA system is given in this Chapter.

4.1 System Architecture

As we described at the beginning of this Chapter, our QA system consists of the document preprocessing component for document normalization and indexing, the question analysis component to identify question type, generate query, and expand the query, the document analysis component for retrieving relevant documents using the queries from the question analysis component and filtering the retrieved documents according to the question type, and the answer extraction component to produce answers from the filtered documents. The whole design of the proposed QA system and the interaction among the components is shown in Figure 4.1.

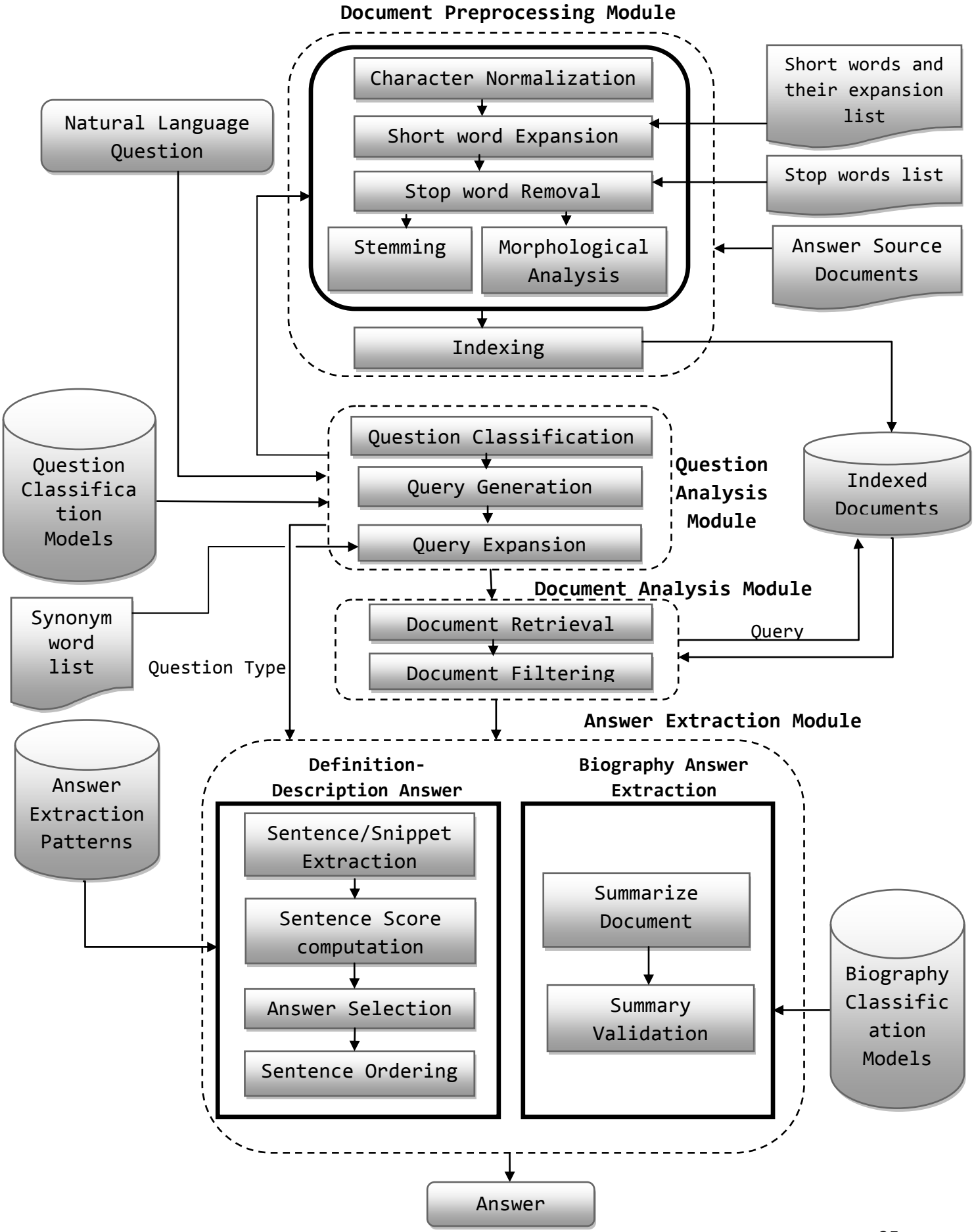

***Figure 4.1**: The Architectural Design of Amharic Definitional, Biographical and Description QA System*

4.2 Document Preprocessing

Amharic is a Semitic language spoken in many parts of Ethiopia and abroad (like in US, Israel, Eritrea, etc.). It is the official working language of the Federal Democratic Republic of Ethiopia and several of the states/regions in our country [35]. The language has its own grammatical and morphological structure [37, 38]. Moreover, it is written using a writing system called fidel or abugida (have more than 380 Unicode representations (U+1200-U+137F)), that are adapted from the Ge'ez language [36].

In Amharic one word can be expressed by different characters (―fidel"). For example, the word ―constitution" could appear as "**ሕገ መንግሥት**", "**ህገ መንግሥት**", "**ሕገ መንግስት**", "**ህገ መንግስት**", "**ኅገ መንግሥት**" and "**ኅገ መንግስት**". This is due to the characters that have the same sound but different structural appearance like **ሀ/ሐ/ኀ**, **ጸ/ፀ**, **ሠ/ሰ**, **አ/ዐ**,… **ሆ/ሖ/ኆ**, **ጸ/ፃ**, **ሥ/ሶ**, and **ኡ/ዩ**. Moreover, a word could have different forms due to the addition of prefixes, infixes, and postfixes, like **ለሕግ**, **ሕግጋት**, and **በሕጋቸው**. The characters that are added on the word could indicate person class, sex, number, or tense. These affixes could change the role of the word to be noun, verb, adjective, adverb, etc. [37, 38]. Also words could be written in short form like "**ሕ/ሰብ**", "**ዓ.ም**" to represent "**ሕብረተሰብ**" and "**ዓመተ ምህረት**" respectively. In addition answer source documents and questions could contain stop words like "**ነው**", "**ናቸው**", "**እና**", etc. Thus to overcome such and other problems we need to perform question and document preprocessing. As [3, 32] showed the ―TETEYEQ" and ―LETEYEQ" QA systems precision was better over the normalized documents than the un-normalized documents.

The document preprocessing component performs character normalization, short word expansion, stop word removal, stemming, morphological analysis to obtain root words, and indexing. Except the morphological analyzer the document preprocessing component that we used is similar to that used in [3, 32].

Character Normalization

This subcomponent is used to normalize characters of the question and answer source documents. The need for character normalization is to avoid any differences on similar words that are written using different characters with the same pronunciation but different structural

appearance. Thus this subcomponent first tokenizes the given text to words, then it splits each word to its characters and inspects if the character is one that is interchangeably usable. If so it will replace it by its normalizing character. Table 4.1 gives the set of characters that can be used interchangeably and the normalizing character for each case.

Table 4.1*: Characters used interchangeably and their normalizing characters*

Characters used interchangeably	**Normalized to**
ሀ፣ ሃ፣ ሐ፣ ሓ፣ ኀ፣ ኃ	ሀ
ሰ፣ ሠ፣ ሱ፣ ሡ፣ … ፣ሶ ፣ሦ	ሰ፣ ሱ፣ … ፣ሶ respectively
ጸ፣ ፀ፣ ጹ፣ ፁ፣ … ፣ጾ ፣ፆ	ጸ፣ ጹ፣… ፣ጾ respectively
ቸ፣ ቼ	ቸ
ች፣ ቺ	ች
ሸ፣ ሼ	ሸ
የ፣ ዬ	የ
አ፣ ኣ፣ ዐ፣ ዓ	አ
ኘ፣ ኚ	ኘ
ሸ፣ ሺ	ሸ
ኰ፣ ኮ	ኮ
ጎ፣ ጐ	ጐ

Short Word Expansion

Documents could contain words written in short forms. Usually in Amharic "." and —"/ are used while writing words in short form. For example, if someone asks the question "**ቤ/ን ምንድን ነው**?"(What is church?) the word "**ቤ/ን**" should be expanded to "**ቤተክርስቲያን**" (church) while searching the answer to the question. The expansion is needed because the word could appear in short form within the question and fully written in the answer source document or vice versa. So the short word expansion solves this problem by expanding the words using set of predefined short words and their expansions listed in Appendix A.

Stop word Removal

Words that appear in many sentences or that do not serve to distinguish one sentence from another are referred to as stop words. Since these words have no any relevance during document retrieval they are removed by this subcomponent using predefined stop words listed in Appendix B.

Stemming

Stemming is an activity to find the stem of a word by removing affixes, i.e., it enables to merge morphological variants of a word under a single index entry or its common form. In this thesis, we used the stemming algorithm developed in [49]. According to [39] the stemming algorithm removes those affixes that are usually used for changing the tense, number, gender and case of a word. Furthermore, in the case of removing suffixes with vowels, the last character of the word after the removal of the suffix is changed to ―sdes" (the six order of a character).

There are situations in which stemming should not be done. For example, consider the texts "**አበበ ባልቻ** …" and "**አበበች ጎበና** …", if the names in the two texts are given to the stemmer then it will make the two names "**አበበ**" and "**አበበች**" as "**አበብ**", in which the two different names in the two documents become identical. Thus to overcome this problem a token will be stemmed only if it is not a person or organization name.

Morphological Analysis

Morphological analysis is the segmentation of words into their component morphemes and the assignment of grammatical morphemes to grammatical categories and lexical morphemes to lexemes [50]. Thus the morphological analyzer returns the root of a word and it enables to merge morphological variants of a word under a single index entry or its common form. With the same reason as in the stemming, name of a person and an organization is not analyzed by this unit.

HornMorpho[4] is used for the morphological analysis task. HornMorpho is a Python program that analyzes Amharic, Oromo, and Tigrinya words into their constituent morphemes (meaningful parts) and generates words, given a root or stem and a representation of the word's grammatical structure [50].

[4] http://www.cs.indiana.edu/~gasser/Research/software.html

Indexing

Indexing is a process of representing the main semantic aspects of the documents and queries [1]. Indexing involves the processing of the original data into a highly efficient cross-reference lookup (index) in order to facilitate searching. We used Lucene Indexer to index the preprocessed documents. The index stores statistics about terms in order to make term-based search more efficient. As stated in [32, 40, 41] Lucene's index falls into the family of indexes known as an inverted index.

An object is instantiated from the Lucene Document class to hold every file that is going to be indexed and its fields with their values are added to the Lucene Fields object. Then an IndexWriter object is instantiated to write the Lucene Document object with its necessary Fields into the Lucene index file.

4.3 Question Analysis

The main function of the question analysis component is to understand the question. Thus the question is analyzed in a number of ways in order to understand what the question asks for. Since identifying the semantic type of the question is an important step for extracting the actual answer, question classification subcomponent is responsible for this task. Then the query generator generates the queries from the asked question according to the type. Finally the generated queries are expanded by the query expander subcomponent. Questions except definition, biography and description type will not be further processed.

4.3.1 Question Classification

The question classifier subcomponent determines the type of a question as biography, definition or description. We used rule based and machine learning approaches to determine the type of question.

***Table 4.2**: Question classes, interrogative terms, and class indicative terms*

Class	Interrogative term	Class indicative term
Definition	ምን ማለት ነው፣ ምንድን ነው፣ ምንድን ናት፣ ምንድን ናቸው	ትርጉሙ፣ ትርጓሜው፣ ትርጓሜ፣ ትርጉም፣ ሲተረጎም፣ ስንተረጉመው፣ ፍቺ፣ ሲፈታ፣ ስንፈታው፣ and ፍቺው
Description	ምንድን ነው፣ ምን … አለው/አላት/አላቸው፣ ምን እንደሆነ አብራሩ/አብራሪ/አብራራ/ግለጽ/ግለጭ/ግለጹ፣ ለምን ይጠቅማል/ትጠቅማለች/ይጠቅማሉ	ጠቀሜታ፣ ድርሻ፣ ጥቅም፣ ፋይዳ፣ አገልግሎት፣ ሚና፣ ተግባር፣ አብራራ፣ አብራሪ፣ አብራሩ፣ ግለፅ፣ ግለጭ፣ ግለፁ፣ ፋይዳው፣ አገልግሎቱ፣ ሚናው፣ ተግባሩ፣ ድርሻው፣ ጥቅሟ፣ ፋይዳዋ፣ አገልግሎቷ፣ ሚናዋ፣ ድርሻዋ፣ ተግባሯ፣ ጥቅማቸው፣ ፋይዳቸው፣ አገልግሎታቸው፣ ሚናቸው፣ ድርሻቸው፣ and ተግባራቸው
Biography	ማነው፣ ማናት፣ and ማናቸው	-

The rule based approach identifies type of questions using the algorithm given in Figure 4.2. The algorithm determines the question type by using the interrogative terms of the question and class indicative terms in Table 4.2. For example, given the question "**በኢትዮጵያ ውስጥ ድህነትን ለመቅረፍ የትምህርት ጥቅም ምንድን ነው**?" (~~-~~what is the use of education for eradicating poverty in Ethiopia?") the terms "**ጥቅም**" (use) and "**ምንድን ነው**" (what) indicate that the question is looking for a description.

```
1.Input = question
2.Flag=0; // An indicator whether the question class is found or not.
3.If question contains (“ምን ማለት ነው” ) then
4.          Question_Type = “Definition”;
5.          Flag=1;
6.Else If question contains (one of the definition indicative term and
         “ምንድን ነው”|“ምንድን ናት”|“ምንድን ናቸው”) then
7.          Question_Type= “Definition”;
8.          Flag=1;
9.Else If question contains (“ማነው”|“ማናት”|“ማናቸው” ) then
10.            Question_Type= “Biography”;
11.             Flag=1;
12.    Else If question contains (one of the description indicative term
        and “ምንድን ነው”|“ምን አለው|አላት|አላቸው”| “ምን እንደሆነ አብራሩ|አብራሪ|
        አብራራ|ግለጽ|ግለጭ|ግለጹ”|“ለምን ይጠቅማል|ትጠቅማለች|ይጠቅማሉ”) then
13.              Question_Type= “Description”;
14.              Flag=1;
15.    End If
16.    If Flag==0 and question contains(“ምንድን ነው”|“ምንድን ናት”|“ምንድን
       ናቸው”) then
17.              Question_Type = “Definition”;
18.    Else
19.              Question_Type = “Unkown”;
20.    End If
```

Figure 4.2*: Rule Based Question Classification Algorithm*

The other approach that we designed for question classification is machine learning approach. This approach predicts the question type of questions based on the training models. The training and classifying tool that we used is Support Vector Machine algorithm implementation known as

SVM_light[5] which is developed by Joachim [28]. SVM is basically a binary classifier. It takes a set of input data and predicts, for each given input, which of two possible classes form the output.

Since we have three question classes namely definition, biography, and description, the binary classifier SVM_light cannot be directly used to find the class of a question. Due to this reason we used one class with other class technique to train the classifier, i.e., the svm_learn method of the SVM_Light is trained by preparing three question groups (definition versus biography, definition versus description, and description versus biography) and it constructs three models based on the three grouped data.

The svm_classify module of the SVM_Light tries to find the class of the question by testing it with the three generated training models for a new question. As a result, the svm_classify returns a number that shows to which question type the classifier classifies the given question. Since there are three models, an aggregate of three numbers are generated. If we see a positive result, it means the classification is into the first class, if negative, it is to the second class. Values close to -1 and 1 are indicating the classification is done with higher accuracy whereas, values near to 0 indicate classifications of poor accuracy resulting in classification into an _unknown‘ dass type.

The classifier is trained by preparing the questions as follows:

1. Term frequency Computation: Find the term frequency of each term within a question. Every term and its frequency within a question are written to a single file. For example, for the question "**ደን ማለት፡ ምን ማለት፡ ነው**" the result became **ደን** 1, **ማለት፡** 2, **ምን** 1, and **ነው፡** 1. These terms and their frequencies will be written to a new file in such a way that every term and its frequency are on a new line.
2. Creating category vocabulary: creates a document that holds every question term of a class and their document frequency within that category.
3. Assign term_ID and determine document frequency: assign a term_ID for each term and determine each term document frequency within the group. Then create a new document that contains every term, its id and document frequency for each group.

[5] http://svmlight.joachims.org/

4. Compute weight of each term in every question of the group. The weight is calculated using Equation 4.1, which is taken from [32].

$$weight(term) = \log\left(tf * \left(\frac{N}{df}\right)\right) \qquad (4.1)$$

where: tf is term frequency, i.e., the number of a term in a question, N is the total number of questions in both classes (the group), and df is the number of documents in a group that contains the term.

Since SVM_Light would be used only if the training or test data is in numeric form, the training statements are prepared in such a way that SVM_Light could accept. The structure is:

—ClassLabel (1 if the question is in first class of the group or -1 if the question is in second class of the group) $Term_1$_ID:weight($term_1$) $Term_2$_ID:weight($term_2$) … $Term_n$_ID:weight($term_n$).” Where n is the number of terms in the question and term_ID is the ID given for each question in that group.

Let us consider the questions "**መሰረተ ልማት ምን ማለት ነው**" and “**አትሌት ደራርቱ ቱሉ ማናት**" in group one (definition versus biography). The statements created for these two questions are:

```
1   39:1.6094379124341003   5:3.9318256327243257   41:3.9318256327243257
20:2.302585092994046 22:1.6094379124341003 and
```

```
-1   59:2.0794415416798357   54:3.9318256327243257   72:2.833213344056216
97:3.9318256327243257
```
respectively.

The generated statements are then sorted by their id and stored by the names *SortedWdefVSbio.txt, SortedWdefVSdes.txt and SortedWdesVSbio.txt.* Thus the sorting result of the above two statements become:

```
1   5:3.9318256327243257   20:2.302585092994046   22:1.6094379124341003
39:1.6094379124341003 41:3.9318256327243257 and
```

`-1 54:3.9318256327243257 59:2.0794415416798357 72:2.833213344056216 97:3.9318256327243257` respectively.

5. Train the classifier: the following command is written to train the SVM_Light tool:

   svm_learn SortedWdefVSbio.txt TrainDefvsBio.model, svm_learn SortedWdefVSdes.txt TrainDefvsDes.model and svm_learn SortedWdesVSbio.txt TrainDesvsBio.model where, svm_learn is an SVM key word to train, *SortedWdefVSbio.txt, SortedWdefVSdes.txt* and *SortedWdesVSbio.txt* are the sorted weight files created at step 4, and TrainDefvsBio.model, TrainDefvsDes.model, and TrainDesvsBio.model are the learnt models created by svm_learn. These models are used while classifying a test question.

Test question preparation and classification using the classifier

Character normalization and short-word expansion is performed on the test question. Then the following steps are followed to prepare the question for the classification. These steps are:

1. Compute term frequency: the frequency of each term in the test question is computed.
2. Determine term_ID and document frequency: For each term in the test question, if the term is found in a group, then its ID is taken from the training set and its document frequency becomes document frequency in the training set + 1. If the term is not found in the training set, a unique new term_ID is generated for it and its document frequency becomes 1.
3. Compute weight: using the calculated document frequency in step 2 and term frequency in step 1, the weight of every term in the test question is calculated using Equation 4.1.
4. Generate and sort the test document: using the identified term_ID and computed weight, the statement is generated as:

   Class_Label (since the class of the question is not known its label is 0) $term_1$_ID:weight($term_1$) $term_2$_ID:weight($term_2$) … $term_n$_ID:weight($term_n$).

   The generated statement is sorted using the term_ID. Since there are three groups, three statements are generated.

   For example, if the test question is "**ዩኒቨርሲቲ በአንድ ማህበረሰብ ውስጥ ያለው ፋይዳ ምንድን ነው?**", then the generated and sorted documents by comparing with definition versus biography, definition versus description and description versus biography are:

0 10:0.6931471805599453 14:2.5649493574615367 17:2.833213344056216 115:3.970291913552122 116:3.970291913552122 117:3.970291913552122 118:3.970291913552122,

0 10:0.0 14:0.6931471805599453 17:2.772588722239781 53:2.1972245773362196 70:2.4849066497880004 74:1.791759469228055 75:2.1972245773362196, and

0 4:0.6931471805599453 8:0.6931471805599453 9:2.5649493574615367 11:2.0794415416798357 30:2.302585092994046 35:1.6094379124341003 37:2.0794415416798357 respectively.

5. Identify question class: By employing the generated classifier models from the training phase and the sorted weights in step 4, call the SVM's classify method as:

   svm_classify testVsdefbioSorted.txt TrainDefvsBio.model group1_result, svm_classify testVsdefdesSorted.txt TrainDefvsDes.model group2_result, and svm_classify testVsdesbioSorted.txt TrainDesvsBio.model group3_result. where, *svm_classify* is and SVM command, *testVsdefbioSorted.txt, testVsdefdesSorted.txt and testVsdesbioSorted.txt* are the sorted weight data the test question with respect to the three groups, TrainDefvsBio.model, TrainDefvsDes.model and TrainDesvsBio.model are the classifier model developed at the end of the training phase, and *group1_result, group2_result, and group3_result* are files to hold the final classification result.

The results obtained for the testing question is given in Table 4.3.

Table 4.3: *SVM classify Groups result for the test question*

No	Group Name	Group Result	Result Indication
1	Group1 (definition vs. biography)	0.53202085	Definition
2	Group2 (definition vs. description)	-0.72334308	Description
3	Group3 (description vs. biography)	0.88033068	Description

After obtaining the results the count of each class out of the three values is used to determine the question class, i.e., the one with higher frequency is supposed to be the question type for the test question. Otherwise, if there is no one class with highest frequency, the question type is decided as ‗unknown'. Thus the question type for the given test question becomes ―Description".

4.3.2 Query Generation

Question analysis also extracts the information in the question that allows the generation of queries that could be used by an IR system for the extraction of the answer-bearing text from the entire document collection. The query generator acquires queries by removing the interrogative terms from the question.

First the characters of the question are normalized, short words are expanded, and stop words are removed, then the query generator removes the interrogative terms that are found within it. The interrogative terms are listed in Table 4.2. Finally it passes the remaining part of the question as a query for the query expansion subcomponent. For example, if the question is "**አምስተኛው የኢትዮጵያ ፓትርያርክ የነበሩት ብጹዕ አቡነ ጳውሎስ ማናቸው**" the query generator result becomes "**አምስተኛው የኢትዮጵያ ፓትርያርክ ብጹእ አቡነ ጳውሎስ**".

4.3.3 Query Expansion

The main problem faced when performing document retrieval for definition and description questions is the lack of terms which can be used as an IR query. Unlike factoid questions which can contain a number of relevant terms (for example, "**በኢትዮጵያ ውስጥ ትልቁ ተራራ የቱ ነው?**" –What is the highest mountain in Ethiopia?" contains the words "**ትልቁ**" —highest", "**ተራራ**" –mountain", and "**ኢትዮጵያ**" —Ethiopia") which can be used to select relevant documents. But definition and description questions consist usually just the target of the definition or the description. That is, when asked to define "***ሀገር***" (–country") the input to the document retrieval will be only "***ሀገር***" (–country"). Due to this reason query expansion is performed. Query expansion is the process of reformulating a seed query to improve document retrieval performance, i.e., it is a way to include documents which have the potential to be more relevant to the question.

The query expansion subcomponent tokenizes the query and checks if a token has a synonym word in the synonym word list, if so it appends the synonym word to the query. Since getting synonym name for an organization and a person is difficult, query expansion is only performed for definition and description question types. Finally the expanded query is analyzed by the morphological analyzer and sent to the document analysis component.

4.4 Document Analysis

Whatever QA architecture is chosen, answering questions over a closed corpus or even the web almost always involves some kind of searching for and retrieval of documents to narrow the search space for the answer to the question [10]. The document analysis component consists of the document retrieval and document filtering subcomponents. That is, the document retrieval is responsible for locating documents which may contain information pertinent to the definition or description of a target. Many systems locate relevant sentences by first retrieving documents which contain any of the words in the target. This is guaranteed to find all documents which discuss the target but will of course find many which have no relation to the target. Thus the retrieved documents are filtered by the document filtering subcomponent.

4.4.1 Document Retrieval

Once the type of question is identified and the queries have been created and expanded, relevant documents are retrieved from the indexed documents by the document retrieval sub component. We used the open-source Java library with several extensions and useful features, IR-system Lucene[6] from the Apache Jakarta project. That is, Lucene is responsible for returning ranked candidate answer bearing documents from the Lucene index. As a result, the set of related documents to the given query will be returned as Hits.

4.4.2 Document Filtering

The document retrieval subcomponent simply ranks and extracts documents based on word overlap to the query; it does not consider the question type. That means a document ranked to top might be irrelevant to a question. Due to this reason documents should be filtered before further analysis. Thus document filtering subcomponent first locates the question target using target extractor and based on the target filters the documents by the document filtering unit.

Question Target Extractor: extracts the target term(s) of the question. If the question type is

- Definition, the query term is taken as the target,

[6] http://jakarta.apache.org/lucene/

- Description, the target is obtained by removing descriptive indicator terms like “**ጥቅም፣ ፋይዳ፣ አገልግሎት፣ ሚና፣ ተግባር፣ ሚናው፣ ጥቅሙ፣ ፋይዳው፣ አገልግሎቱ፣ ድርሻው፣ ተግባሩ፣ ጥቅሟ፣ ፋይዳዋ፣ አገልግሎቷ፣ ሚናዋ፣ ድርሻዋ፣ ተግባሯ፣ ጥቅማቸው፣ ፋይዳቸው፣ አገልግሎታቸው፣ ሚናቸው፣ ድርሻቸው፣ ተግባራቸው**” from the query, and
- Biography, the target becomes the query itself if the question focus is an organization. On the other hand if the focus is a person, the query will be examined whether it contains the titles listed in Table 4.4, and then the terms with and without the title are taken as target otherwise the query will be taken as the target. The question focus is determined by the Named Entity Recognizer obtained from the work of [3]. First the query terms are tokenized and passed to the NER, if one of the tokens is identified as organization then the focus becomes organization; and if two or more of the terms are identified as person then the focus becomes person.

***Table 4.4**: Titles for person name*

Titles
አቶ፣ ወ/ሮ፣ ወይዘሮ፣ ወ/ሪት፣ ወይዘሪት፣ ዶ/ር፣ ዶክተር፣ ሸህ፣ ሸክ፣ ቄስ፣ ክቡር፣ ክብርት፣ ሻምበል፣ ሻንበል፣ ኮሎኔል፣ ኮሎኔል፣ አስር አለቃ፣ አ/አለቃ፣ አምሳ አለቃ፣ ሻለቃ፣ ጀኔራል፣ ጀነራል፣ ፕሮፌሰር፣ ፕ/ር፣ ወ/ር፣ ወታደር፣ ኢንጅነር፣ ድያቆን፣ ባላምባራስ፣ ብላቴን ጌታ፣ ፊታውራሪ፣ ብላታ፣ አባ፣ ደጃዝማች፣ ሜጀር ጀነራል፣ በጅሮንድ፣ መምህር፣ ግራዝማች፣ ሊቀ ጠበብት፣ ነጋድራስ፣ ልኡል፣ ራስ፣ አቡነ፣ መምህር፣ አለቃ፣ ብላታ፣ ሀኪም፣ ነጋድራስ፣ ሀጂ፣ አርቲስት፣ አፈ ጉባኤ፣ የተከበሩ፣ አምባሳደር፣ ኮማንደር፣ ብርጋድየር ጀኔራል፣ ሌተናል ኮሌኔል፣ ሹም፣ አፄ፣ መቶ አለቃ፣ ሚስተር፣ ጠ/ሚ፣ ሚኒስትር ድኤታ፣ ብፁእ፣ ተመራማሪ፣ ከንቲባ፣ ሊቀመንበር፣ ምክትል፣ ሳጅን፣ ሎሬት፣ አሰልጣኝ፣ አምበል፣ ኡስታዝ፣ ኢንስትራክተር፣ ሰአሊ፣ ፒያኒስት፣ ጠቅላይ ሚኒስትር፣ ሚ/ር፣ ጠ/ሚኒስትር፣ ፕሬዝዳንት፣ ፕረዝዳንት፣ ፕሬዚዳንት፣ ፕሬዚደንት፣ ፕረዝደንት፣ ካፒቴን፣ ፓትሪያርክ፣ ፕ/ት፣ እነ፣ ዋና ዳይሬክተር፣ ዳይሬክተር፣ ኢንስፔክተር፣ አትሌት

After the target is extracted, for description and definition question types, the documents will be tested by their respective rules (the rules are formed as regular expressions) listed in Table 4.5. Then, if a text is extracted by one of the rules, the document will be kept; otherwise it will be removed.

For biography question types, the documents that contain all the target terms in the same order as in the query will be taken for further analysis, i.e., the document should contain the name of the person or the organization because one term and terms order difference on person‘s or organization‘s name could refer to other person or organization. Moreover, in this research we have discovered that most biography documents‘ first sentences contain the name of the person or the organization under discussion. Thus, the document filtering subcomponent retains documents that contain only the target in their first line.

4.5 Answer Extraction

Answer extraction is the act of extracting text strings constituting the exact answer or answers to a question from the candidate answer-bearing document(s) retrieved by the document analysis module. To identify relevant information more accurately, the answer extraction module performs detailed analysis on the retrieved documents and pin-points the answer to the question as specified by the question analysis component, i.e., according to the question type, the filtered documents will be processed by the biography answer extraction or definition-description answer extraction subcomponents.

4.5.1 Definition-Description Answer Extraction

Factoid questions require a single fact (or a set of facts for a list question) to be returned to the user. So the two factoid Amharic QA works in [3, 32] extract answers using the named entity (gazetteer) and pattern based answer pinpointing algorithms. These algorithms only identify person or place name, dates, and numbers. But definition and description questions require a substantially more complex response – a short paragraph which succinctly defines the target or state concepts the user wishes to know more about. That means the methods in [3, 32] cannot be used for biography, definition, and description questions. Thus finding snippets or piece of information about the current target, ranking, selecting, and ordering them is very important. To

do so, the definition-description answer extraction subcomponent comprises snippet/sentence extraction, sentence score computation, answer selection, and sentence ordering units.

Sentence/Snippet Extraction

The main purpose of this component is to create candidate answer set. First every filtered document is tokenized to sentences by the sentence tokenizer. The sentence tokenizer splits the given document into its sentences using "።/።/::" characters as delimiter. Then according to the question type the snippet/sentence extractor extracts sentences from the tokenized sentences using manually crafted indicative patterns that are listed in Table 4.5. About 14 rules (regular expressions) for definition and 5 rules for description questions are crafted by inspecting different Amharic definition and description bearing documents. We observed that many documents, while defining a term they usually state the core concepts about the term at the beginning. Due to this reason, for definition questions, if the first sentence of a document is extracted by one of the rules, the next two sentences are incorporated to the candidate answer set.

***Table 4.5**: Sentence/Snippet Extraction Patterns*

Question Type	**Sentence Extraction Patterns**
Definition	Rule 1: target+" ማለት " +".*" Rule 2:target+"(ማለት)?"+".*"+" ማለት ነው[::|።|::]" Rule 3: target +" .* ትርጉ[ሙ|ም]|ፍቺው (.*|ማለት ነው[::|።|::])" Rule 4: ".*"+target +".*"+" ወይም "+".*" Rule 5: ".*"+ " ወይም "+target+" .*" Rule 6: ".*"+ target+" .*"+"(ሊተረጎም| ሊፈታ| ሊጠራ| ነው ማለት)?"+"(ይችላል| ይቻላል ([::|።|::]))" Rule 7: target + " .*"+"(ሲል| ሲሆን| ሲባል)"+" .*" Rule 8: ".* " + target +" ይባላል[::|።|::]" Rule 9: target +" .* "+"(በመባል |ተብሎ| ተብሎም)?"+ "(ይታወቃል| ይተረጎማል| ይፈታል| ይጠራል([::|።|::])" Rule 10: target +" .*"+"(ናት]| ናቸው| ነው([::|።|::])"

	Rule 11: target+" .* "+"ነው ማለት ይቻላል[::|።|፡፡]" Rule 12: ".*"+" ማለት "+".* " +"([በ|ከ|የ])?"+target+ " .*" Rule 13: ".* "+target+" ማለት "+".* " Rule 14: ".*"+"\\("+target+"\\)" +".*"+ "(ማለት)?"+".*"
Description	Rule 1: target+" .*"+"(ጥቅ[ም|ሙ|ሞች|ሙም|ማቸው] | ሚና[ውም|ው|ዎች|ዎቹ |ዎቿ|ቸው] |ድርሻ[ም|ሙ|ሞች] | አገልግሎት[ቱ|ቶች]|ተግባ[ር|ሩ|ራት])"+".*" + "(ያለው | ያላቸው | ያላት)?"+ ".*"+"(ይውላል[::|።|፡፡]| ትውላለች[::|።|፡፡]| ይውላሉ |አላት |አላቸው | አለው | አሉት | ይኖሩታል | ይሰጣል | ናት | ናቸው | ነው([::|።|፡፡|፡፡]))?" Rule 2: target+" .*"+"(ጥቅ[ም|ሙ|ሞች|ሙም|ማቸው] | ሚና[ውም|ው|ዎች|ዎቹ|ዎቿ|ቸው] |ድርሻ[ም|ሙ|ሞች] | አገልግሎት[ቱ|ቶች|ቶቹ|ቶቿ|ታቸው]|ተግባ[ር|ሩ|ራት])?"+ ".*" + "(ያለው | ያላቸው| ያላት)?"+ ".*"+"(ይውላል[::|።|፡፡]| ትውላለች[::|።|፡፡]| ይውላሉ |አላት |አላቸው | አለው | አሉት | ይኖሩታል | ይሰጣል | ናት | ናቸው | ነው([::|።|፡፡|፡፡]))" Rule 3: target+" .*"+"ተግባር (ሊጠቅም|ሊያገለግል)?(ይችላል |አላት |አላቸው | አለው | አሉት([::|።|፡፡]))" Rule 4: target+".*"+"(ስለሚጠቅም | ይጠቅማል| ያገለግላል([::|።|፡፡]))" Rule 5: target+" .*"+"(የሚጠቅም| የሚያስችል| የሚጠቅሙ| የሚያስችሉ| የምታስችል| የምትጠቅም)"+".*"+"(ናት| ናቸው| ነው([::|።|፡፡]))"

Sentence Score Computation

An answer to a definition or description question should contain all the vital snippets or sentences. Thus, in order to select the appropriate sentences from the candidate answer set we formulate the sentence scoring function given in Equation 4.2, i.e., the score of a sentence S is calculated as the sum of the percentage of the query (Q) terms in the sentence, weight of the pattern that identifies the sentence, the reciprocal of the position of the sentence in the document that contains it, and the Lucene score of the document D that contains S.

$$score(S) = \frac{N_{S \cap Q}}{N_Q} + weight(S, P) + \frac{1}{pos(S)} + luceneScore(D, S) \quad (4.2)$$

Where $N_{S \cap Q}$ is the number of terms that are found in both S and Q, N_Q is the number terms in Q, $weight(S, P)$ is the weight of the pattern P that matches with S, $luceneScore(D, S)$ is the score of document D that contains S by Lucene, and $pos(S)$ is the position of S in the document that contains S.

Since the position of a sentence does not have any impact for description questions, score of sentence S is computed by the formula given in Equation 4.3.

$$score(S) = \frac{N_{S \cap Q}}{N_Q} + weight(S, P) + luceneScore(D, S) \quad (4.3)$$

Answer Selection

At this stage this subcomponent faced a set of scored text fragments which are possibly used to generate the correct answer. But it is likely that some of the snippets or sentences might be redundant and all are not equally important. That is, we have to guarantee that, given sentence A does another sentence B provide any new and relevant information for the purpose of defining or describing a given target? Therefore, we need an approach that would determine the semantic similarity of the extracted snippets/sentences and their importance to the target relative to each other. As the work in [26] suggested one way of determining the similarity of texts is to use word overlap. The more different text fragments share common non-stop words it indicates that they are highly similar [12]. A sentence profile is constructed for each sentence which contains the set of non-stop words, T, in the sentence. Then the similarity, sim, between sentences A and B is calculated by the formula given in Equation 4.4 which is adopted from [26]. This is the percentage of tokens in A which appear in B or the percentage of tokens in B which appear in A.

$$sim(A, B) = \frac{|T_A \cap T_B|}{\min(|T_A|, |T_B|)} \quad (4.4)$$

where $sim(A, B)$ is the similarity of the sentences A and B, $|T_A| \; and \; |T_B|$ are the number of non-stop tokens in sentences A and B respectively, and $|T_A \cap T_B|$ is the number of common tokens in A and B.

Thus, in order to construct the final answer the answer selection subcomponent ranks the sentences by their score, selects the top ranked definition/description sentences according to the definition/description length requirement and avoids introducing any redundant sentences into the result. Figure 4.3 shows the algorithm that we used for answer selection. After consulting different definition and description documents we determined the number of sentences to incorporate in the final answer to be 5.

1. Input: set of sentences in the candidate answer set with their respective score.
2. Sort all sentences in the candidate answer set in descending order by their score.
3. Add the first sentence to the definition/description pool.
4. Examine the *similarity* of the next sentence S in the remaining sentences to all sentences already in the definition/description pool using Equation 4.4.
5. If *the similarity of the sentence S to one of the sentences in the definition/description pool is greater than or equal to 0.7,* then skip sentence S; otherwise add it to the definition/description pool. (The value of the similarity cutoff point is set to 0.7 by doing an experiment).
6. Repeat from Step 3 until the desired number of definition/description sentences is reached to 5.
7. Output: top 5 non redundant sentences.

Figure 4.3*: Answer Selection Algorithm*

Sentence Ordering

Displaying the result of the answer selection as it is might affect the coherence of the answer. Thus this subcomponent orders the sentences in a way that, sentences that begin with the target term will be positioned to the beginning, sentences that begin with connective terms like "ስለሆነም", "በመሆኑም", "በተጨማሪም", "ይህም", "ይህን", "ስለዚህ", "እንዲሁም", "ከዚህ …", "ነገር ግን", etc. will be in the middle, and sentences which start with other terms position will be after the others. The sentences score is used for ordering sentences that have the same priority.

4.5.2 Biography Answer Extraction

For biography questions whose focus is a person, important dates in their life (birth, marriage, and death), their major achievements and any other interesting items of note would comprise a _œorrect‘ answer. Otherwise if the focus is an organization, the answers should probably include information about when and by whom the company was founded, what the company makes/sells, who owns the company, and other interesting things such as other companies they have bought or collaborated with [26]. Therefore, to generate an answer as stated the biography answer extraction component first merges the filtered documents, summarizes the merged document, and validates the summary. Finally, if the result of the validation is affirmative, the summary is displayed as an answer.

Summarize Document

Each relevant filtered document will be merged as one document and summarized by the summarizer of [39]. As discussed in [39] briefly, the summarizer determines the best candidate sentences for the summary generation using different sentence ranking approaches.

Summary Validation

The result of the summarizer is evaluated whether it contains basic information that a biography should have. The evaluation task can be considered as a text categorization problem. To do so we designed a text categorizer that categorizes the result of the summarizer as biography or other. Different biographical and other texts are collected to train the SVM_Light (the SVM algorithm implementation) and use the model to classify the summary produced by the summarizer.

The training documents are first normalized by the document preprocessing components. Then a number of training preparation activities are done on it before they are passed to the svm_learn method. These activities are the following:

1. Term frequency Computation: Find the term frequency of each term within a training document. Every term and its frequency within a document are written to a single file. These terms and their frequencies are written to a new file.
2. Creating category vocabulary: creates a file that holds every training document terms and their document frequency.

3. Assign term_ID and determine document frequency: assign a term_ID for each term and determine each term document frequency. Then create a new file that contains every term, its id and document frequency.
4. Compute weight of each term in every training document and sort. The weight of every term is calculated using the formula given in Equation 4.1. Then the generated statements will be sorted by their id. Since SVM_Light would be used only if the training or test data is in numeric form, the training statements are prepared in such a way that SVM_Light could accept. The structure is:

   ―ClassLabel (1 if the document is biography or -1 if the document is not biography) $Term_1$_ID:weight($term_1$) $Term_2$_ID:weight($term_2$) … $Term_n$_ID:weight($term_n$).” where n is the number of terms in the document and term_ID is the ID given for each term in the document.

5. Train the classifier: the svm_learn method will generate and store the training model on the specified location using the documents generated in step 4.

On the other hand, after normalizing the summarizer result by the document preprocessing components the following activities are performed to prepare the summary (the result of the summarizer) for the classification. These are:

1. Compute term‘s frequency: the frequency of each term in the summary is computed.
2. Determine term_ID and document frequency: For each term in the summary, if the term is found in the training, then its id is taken from the training set and its document frequency becomes document frequency in the training set + 1. If the term is not found in the training set, a unique new term_ID is generated for it and its document frequency becomes 1.
3. Compute weight: using the calculated document frequency in step 2 and term frequency in step 1, the weight of every term in the summary is calculated using Equation 5.1.
4. Generate and sort the test statements by term_ID: using the identified term_ID and computed weight, the statement is generated as;
   Class_Label (since the class of the summary is not known its label is 0) $term_1$_ID: weight ($term_1$) $term_2$_ID: weight ($term_2$) … $term_n$_ID: weight ($term_n$), and sort by the term id.

5. Identify the summary class: call the SVM‘s classify method svm_classify by passing the generated classifier model from the training phase and the summary sorted weights in step 4 as parameter.

The summary validation subcomponent returns the summarizer result as answer for the question if the result of the classifier is greater than or equal to 0.5, otherwise no answer will be displayed. We observed that documents that obtain values greater or equal to 0.5 by the classifier contain the basic information that a biography should contain. Thus we choose the cutoff point to be 0.5.

In general, after passing all steps according to the question type, the final answer for the posed question is displayed for the user as shown in Figure 4.4.

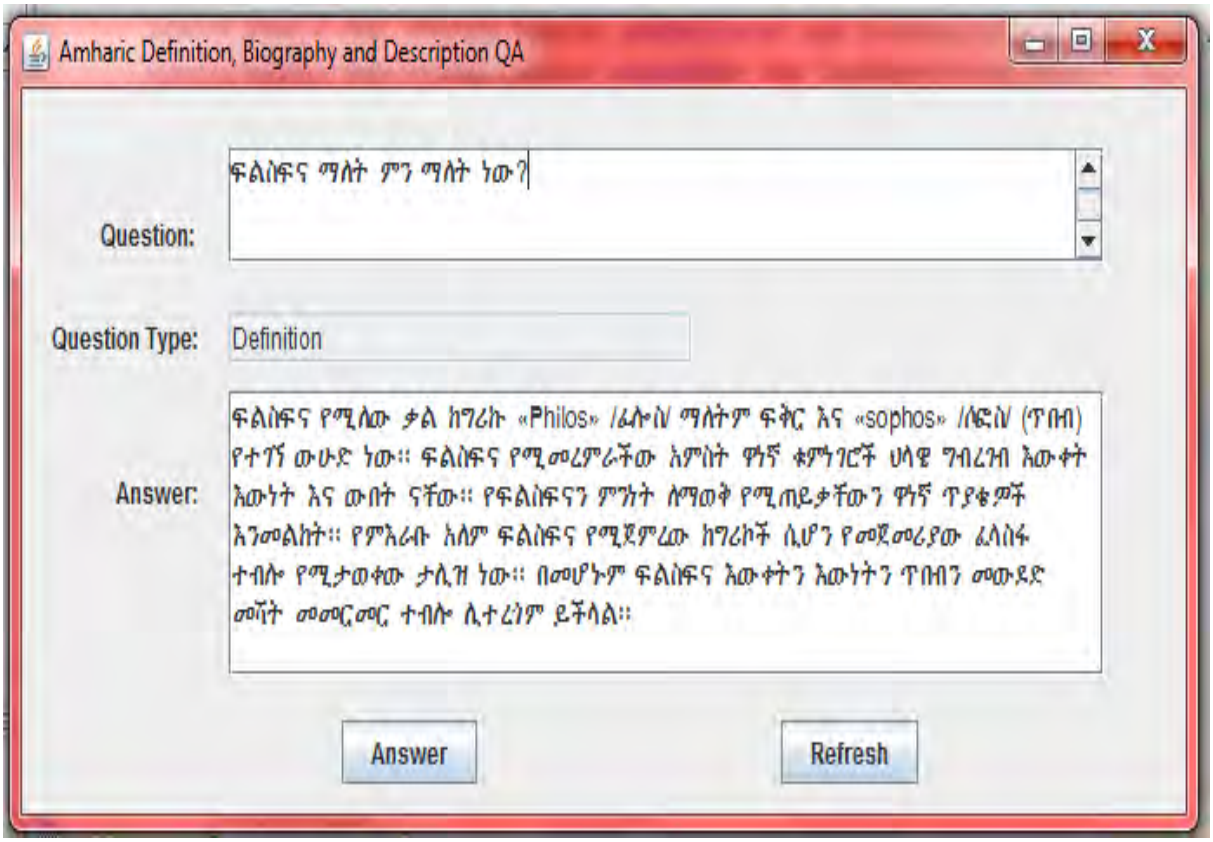

***Figure 4.4**: Prototype User Interface*

4.6 Summary

This chapter described the architectural design of the Amharic definitional, biographical and description QA system and the implementation of its main components. The QA system comprises of four main components. The document pre-processing component is responsible for performing answer source document and question normalization. The question analysis component is dedicated to process the accepted question and determine its question type, generate query and expand the query. Rule based and machine learning techniques are used for the question classification. The document analysis component retrieves relevant documents using the query from the question analysis component and filters the retrieved documents using filtering patterns for definition and description questions and for biography questions a retrieved document is only retained if it contains all terms in the target in the same order as in the question.

The answer extraction component works in type-by-type manner and it is dedicated for answer retrieval. That is, according to the question type the filtered documents are processed by the definition-description or biography answer extraction sub components. The definition-description answer extractor first extracts sentences using manually crafted answer extraction patterns by splitting the document to sentences. Their score is computed by the sentence scoring subcomponent. Then the answer selection algorithm selects top 5 non-redundant sentences from the candidate answer set. Finally the sentences are ordered by the sentence ordering subcomponent and returned to the user. The biography answer extractor summarizes the filtered documents by merging them, and then the summary is displayed as an answer after it is validated by the summary validation subcomponent.

Chapter Five: System Evaluation and Results

In this Chapter, we describe the experimentation environment, the data set, the evaluation metrics, and the results of the performance of the Amharic definition, description, and biography QA system components. In addition discussion on the methods used, activities followed, and results obtained are presented in detail.

5.1 Experimentation Environment

The prototype was developed using Java programming language. Standard Java classes from *java.io* and *java.util* packages were used in collaboration with Lucene and external Java libraries to perform tasks such as accessing files, dealing with arrays and handling exceptions. The eclipse Java editor has been used to develop our system.

The system is developed and tested on a computer with the following specifications:

- Windows® 7 Ultimate™ operating system
- RAM size 2GB
- Hard disk size 250GB and
- Intel® Core™2 Duo Processor 2.166GHz.

5.2 Evaluation Criteria

Automatic QA evaluation has been studied from a number of different perspectives such as the use of test collections [42], reading comprehension tests [43, 44], and the application of automatic systems that evaluate the correctness of the answers returned by comparing them with human-proposed answers to the same set of questions [45]. Based on these perspectives a wide variety of techniques have been employed. By far the most influential evaluation framework has been provided by the TREC[7] QA track first introduced in 1999 [46].

[7] http://trec.nist.gov

Evaluation for QA system mainly focuses on the accuracy of the answers returned. Thus, as suggested in [45] our QA system‘s accuracy is evaluated by comparing the answers returned by the system with human-proposed answers to the same set of questions. Hence, we used precision, recall, and F-score as criteria for measuring the performance.

Recall (R) gives the fraction of sentences that the system has chosen from the totality of sentences found in the actual answer.

$$R = \frac{|system\ \ and\ human\ \ sentence\ \ choice\ overlap\ |}{|senteces\ \ chosen\ by\ human\ |} \quad (5.1)$$

Precision (P) measures the fraction of system answers that are correctly chosen.

$$P = \frac{|system\ \ and\ human\ \ sentence\ \ choice\ overlap\ |}{|senteces\ \ chosen\ by\ system\ |} \quad (5.2)$$

F-score (F) is the harmonic mean of recall and precision.

$$R = \frac{2*P*R}{P+R} \quad (5.3)$$

On the other hand, since the task of the question classifier is identifying question types, its accuracy is evaluated by the percentage of correctly identified question types. The percentage is computed by taking the ratio of correctly identified questions to the total test questions.

5.3 Question Classification Evaluation

A correct answer for a question would probably be extracted if its question type and the expected answer type are correctly identified. Since this task is performed by the question classifier, its performance should be evaluated.

As we discussed in Chapter 4, we have rule based and SVM based approaches for question type classification. For the evaluation of these classifiers, we have prepared 1500 questions from different Amharic academic books and articles of different Amharic official websites. The detail of the number of questions prepared is given in Table 5.1.

***Table 5.1**: Amount of questions used for training and testing*

Type of Question	No. of Training Question Set	No. of Testing Question Set	Total
Biography	450	50	500
Definition	450	50	500
Description	450	50	500
Total	1350	150	1500

The SVM based classifier is trained and tested using n fold cross validation technique. As discussed in [47], the dataset is randomly reordered and then split into n folds of equal size. On each round one fold is used for testing and the other n-1 folds are used for training the classifier. The test result of the iterations is then collected and averaged over all folds which give the cross-validation estimate of the accuracy.

Thus for the evaluation we have used a 10-fold cross validation technique. That is, ten separate experiments are done by reserving one tenth of the question set (150 questions) for testing and using the rest of the data (1350 questions) for training the classifier in each experiment. Since three classification models are generated in each round a total of 30 models are created in the ten experiments. The average of the ratio of the correctly classified to the total number of testing questions in the ten experiments is taken as the performance measurement of this classifier.

The performance of the rule based classifier is obtained in the same manner as that of SVM based classifier. That is, the experiment is conducted using the 150 test questions, that are chosen from biography, definition, and description (i.e., 50 from each) question types on the iterations of the 10-fold cross validation of the SVM based classifier. The performance of this classifier is also obtained by taking the average of the ratio of the correctly classified to the total number of testing questions.

Tables 5.2, 5.3, and 5.4 show the proportion of correctly classified questions out of 50 questions by both of the question type identification techniques during the ten experiments on biography, definition, and description test questions respectively.

Table 5.2: *Performance of the rule based and SVM based question type classifiers on biography questions*

Experiment No.	Classification Technique	
	SVM based	**Rule Based**
1.	1.0	0.98
2.	1.0	1.0
3.	0.9	0.98
4.	1.0	1.0
5.	0.98	0.98
6.	1.0	0.98
7.	1.0	1.0
8.	1.0	1.0
9.	0.96	1.0
10.	0.98	0.98
Average	0.982	0.99

Table 5.3*: Performance of the rule based and SVM based question type classifiers on definition questions*

Experiment No.	Classification Technique	
	SVM based	**Rule Based**
1.	0.7	0.98
2.	0.6	1.0
3.	0.5	1.0
4.	0.22	1.0
5.	0.92	1.0
6.	0.8	1.0
7.	0.74	1.0
8.	0.6	0.96
9.	0.76	1.0
10.	0.78	1.0
Average	0.662	0.99

***Table 5.4**: Performance of the rule based and SVM based question type classifiers on description questions*

Experiment No.	Classification Technique	
	SVM based	**Rule Based**
1.	1.0	0.98
2.	0.12	1.0
3.	0.98	1.0
4.	0.98	1.0
5.	1.0	1.0
6.	1.0	1.0
7.	1.0	1.0
8.	0.76	0.88
9.	0.94	0.9
10.	0.78	0.86
Average	0.86	0.962

As shown in Tables 5.2, 5.3, and 5.4 on question type by question type evaluation, both question type identifiers have no problem on identifying biography questions. This is due to the reason that either the definition or the description question types do not have any common interrogative term with biography, the SVM and rule based classifiers correctly identify 98.2% and 99% of the questions out of 50 biography questions on average respectively. Out of the 50 definition questions on average the SVM and rule based classifiers correctly identify 66.2% and 99% of the questions respectively. Moreover, they correctly identify 86% and 96.2% description questions respectively with the same number of questions. From these average results we see that the SVM based classifier performance on definition questions is minimal than that of the others. This is because of the interrogative word "**ምንድን ነው**", i.e., it is shared by definition and description question types.

On the other hand, the average performance (proportion of correctly classified questions) of the two classifiers on the 150 test questions (50 from each question type) of the 10 experiments is given in Table 5.5.

***Table 5.5**: Performance of the rule based and SVM based question type classifiers*

Experiment No.	Classification Technique	
	SVM Based	**Rule Based**
1.	0.9	0.98
2.	0.573	1.0
3.	0.793	0.993
4.	0.733	1.0
5.	0.967	0.993
6.	0.933	0.993
7.	0.913	1.0
8.	0.787	0.96
9.	0.887	0.967
10.	0.847	0.947
Average	0.833	0.983

Figure 5.1 also shows the average performance of the question classifiers on each experiment on the 150 (50 questions from each type) test questions.

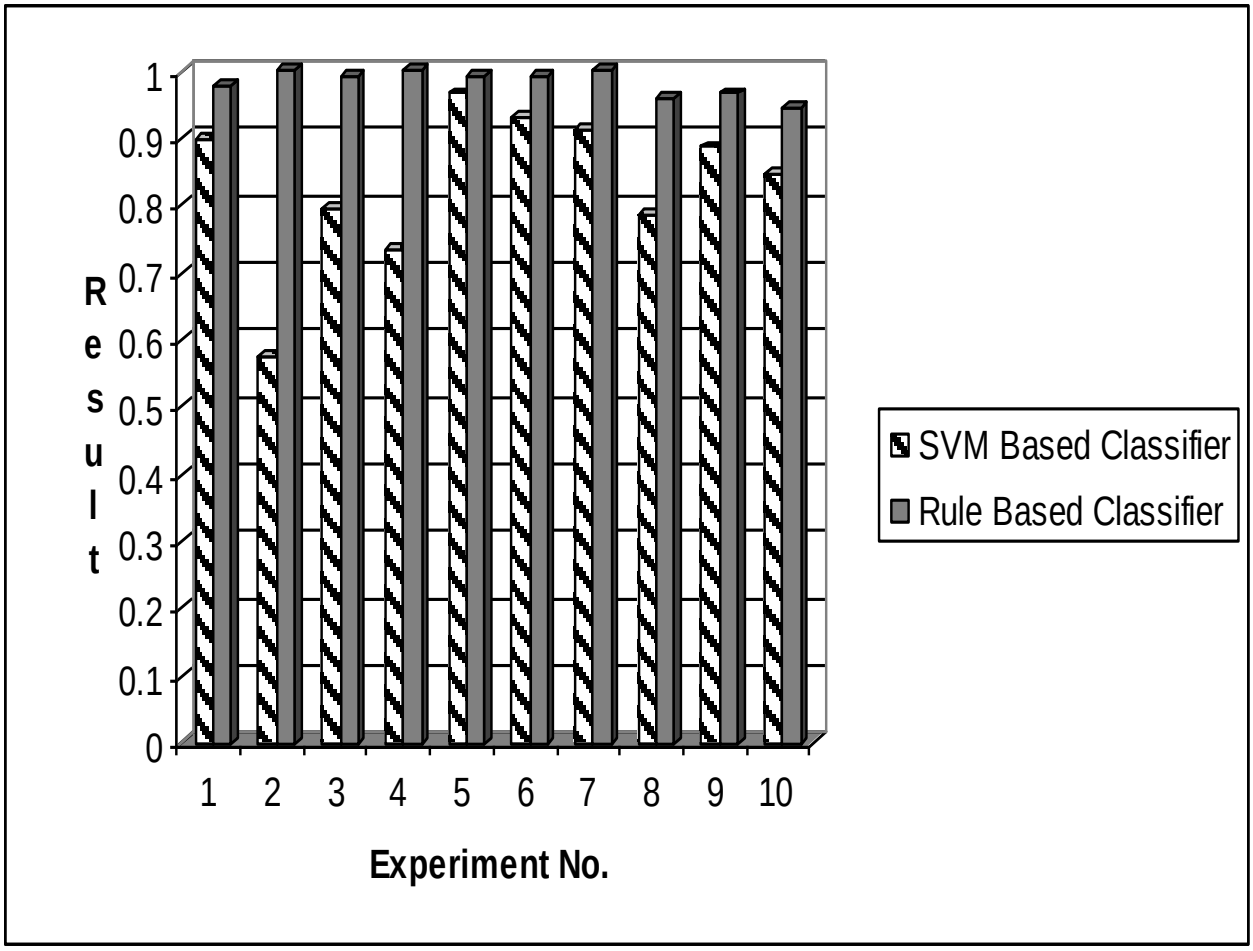

***Figure 5.1**: Question classification evaluation result*

Generally, from the evaluation result on the whole (150 test questions) given in Table 5.5 and shown in Figure 5.1, the average performance of the SVM based question type classifier is 83.3% and that of the rule based question type classifier is 98.3%. That is, the SVM based classifier on average is able to identify the question type of 125 questions correctly out of the 150 test questions and the rule based classifier on average is able to identify the question type of 147 questions correctly out of the 150 (50 questions from each question type under consideration) test questions. This shows that the rule based technique performance is better than that of the SVM based. Sample questions and classification results from the experiment on experiment 10 is attached in Appendix C.

5.4 Document Retrieval Evaluation

In this thesis we measure the search system‘s ability to extract all relevant information from the collection. Information retrieval systems are evaluated with respect to the notion of relevance judgment by human that a document is relevant to a query. In such contexts, we want both high precision (fraction of retrieved items that are relevant) and high recall (fraction of relevant items that have been retrieved). In order to get a synthetic measure from both precision and recall, the harmonic mean between the two (known as F1 or F score) is also used.

According to [48] recall is the ratio of the number of relevant documents returned to the total number of relevant documents in the collection. Of course, a system can achieve 100% recall by simply returning all the documents in the collection. We therefore also need a measure of how many of the documents returned for a given query are actually relevant, which can be assessed by a precision metric. Precision is the ratio of the number of relevant documents returned to the number of documents returned.

The document retrieval performance is measured by 60 queries on two data sets each containing 300 documents. The first data set contains the documents that are analyzed by the stemmer obtained from [3, 32] as they modified the work of [49] (i.e., each document is stemmed), and the second data set contains the same documents as the first data set but they are analyzed by the morphological analyzer of [50] (i.e., each document contains root words).

Then each query is stemmed and analyzed by the morphological analyzer to get the root word. Finally, according to the query search result the recall and the precision are computed. Table 5.6 gives the average recall and precision, and the F-score of our document retrieval component on the two data sets. In addition, to describe the result more, the result in Table 5.6 is presented by the chart in Figure 5.2.

Table 5.6*: The recall, precision, and F-score of the document retrieval component on the two data sets*

Data set	**Recall**	**Precision**	**F-score**
Data Analyzed by the Morphological Analyzer	0.848	0.695	0.764
Data Analyzed by the Stemmer	0.831	0.65	0.729

As Table 5.6 and Figure 5.2 show the document retrieval performance on the stemmed documents is lower than the data set analyzed by the morphological analyzer. Since the morphological analyzer generates root words, the document retrieval has better performance on the lemmatized documents than the stemmed documents.

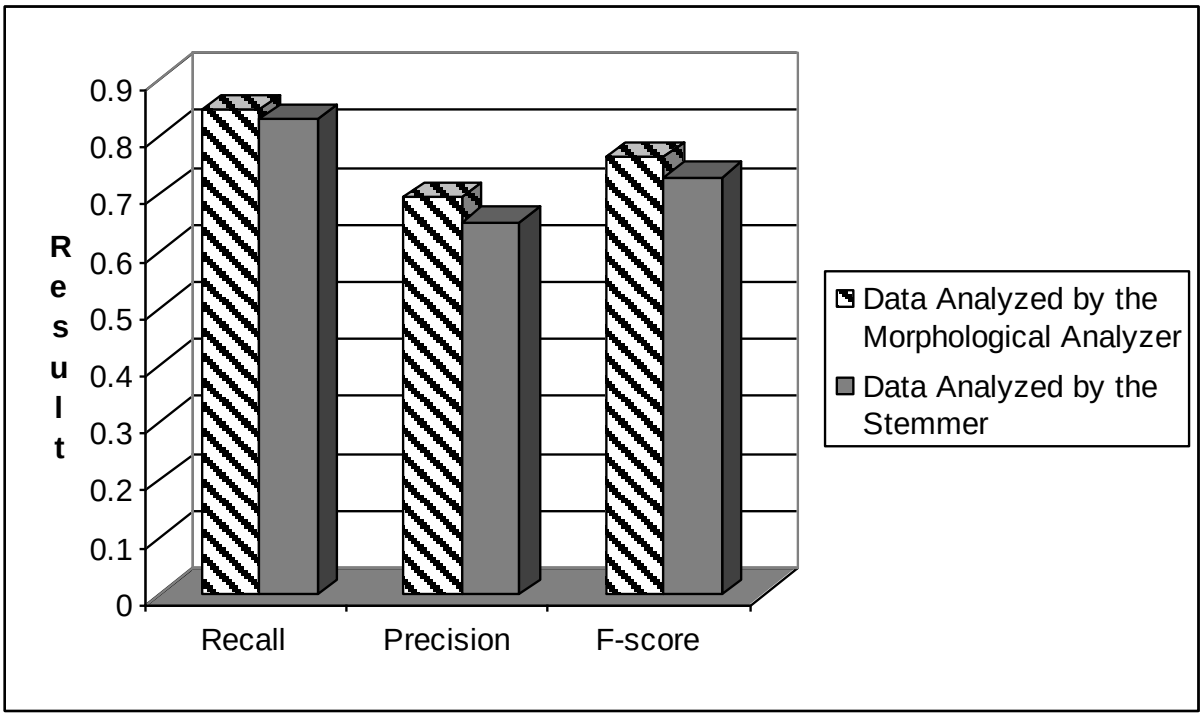

Figure 5.2*: Document Retrieval performance on stemmed and root word data sets*

During our experiment we observed that both the stemmer and the morphological analyzer didn‘t generate the stem and root of every term in the test documents and queries. Moreover, when the morphological analyzer returns root words it gives in Romanized form, for some words in an attempt to transliterate to Amharic the result contains non Amharic characters. For example, the root returned for the term ―ኣይን‖ is ―ነ*‖ when it is transliterated the result is ―ኣይ*‖. Thus the performance of the stemmer and the morphological analyzer should be improved.

On the other hand the stem returned for the terms ―የበራ‖,―ፀር‖ and ―ፀሬ‖ is ―ፀር‖. Likewise the root word of ―ኣይን‖ and ―ኣስተያየት‖ is ―ኣይ*‖. In which the terms ―የበራ‖,―ፀር‖ and ―ፀሬ‖ are different concepts, the same as the terms ―ኣይን‖ and ―ኣስተያየት‖ are also some how different.

Even if the stemmer and the morphological analyzer have advantage on maximizing the document retrieval performance, they could be a cause for retrieving unrelated concepts to the query. As a result this demands document filtering.

5.5 Answer Extraction Evaluation

Biography, definition, and description questions are more difficult to evaluate than factoid questions in that their answers are more complex and can be expressed in a very large number of different forms. In particular, since answers are longer (e.g., longer phrases or sentences), paraphrasing, the use of different surface forms, syntax and semantics play a much more prominent role than in answers to factoid questions. Moreover, answers to these questions can focus on different facets of the concept, entity, person, or organization they are describing, adding to biographical, definitional, and description question complexity.

For the evaluation of the answer extraction component a total of 120 questions are prepared. The questions are composed of 40 biography, 40 definition, and 40 description questions. The answers for these questions generated by our system and the answers that are manually constructed are used to compute the precision, recall, and F-score of each question using Equations 5.1, 5.2, and 5.3 respectively. The average of the precision, recall, and F-score values of each question group is given in Table 5.7 and in Figure 5.3.

Table 5.7*: Recall, Precision, and F-score result of the Answer Extraction component*

Question Type	Recall	Precision	F-score
Biography	0.341	0.895	0.493
Definition	0.725	0.626	0.658
Description	0.762	0.528	0.624
Average	0.609	0.683	0.592

As shown in Table 5.7, even if the difference is minimal the F-score obtained for definition question is greater than the F-score of the description questions. This shows that our answer extraction patterns for definition questions are better than that of the description answer extraction patterns. While conducting the experiment we observed that many documents do not explicitly put the purpose of concepts/entities by using descriptive implication terms like "ዱርሻ፣ ጥቅም፣ ፋይዳ፣ አገልግሎት፣ ሚና፣ ተግባር" rather they put it implicitly. Even in some documents the descriptions are incorporated within their definitions. Due to these reasons the F-score on description questions is less than that on the definition. On the other hand the result that we obtained for the biography questions is highly dependent on the performance of the summarizer.

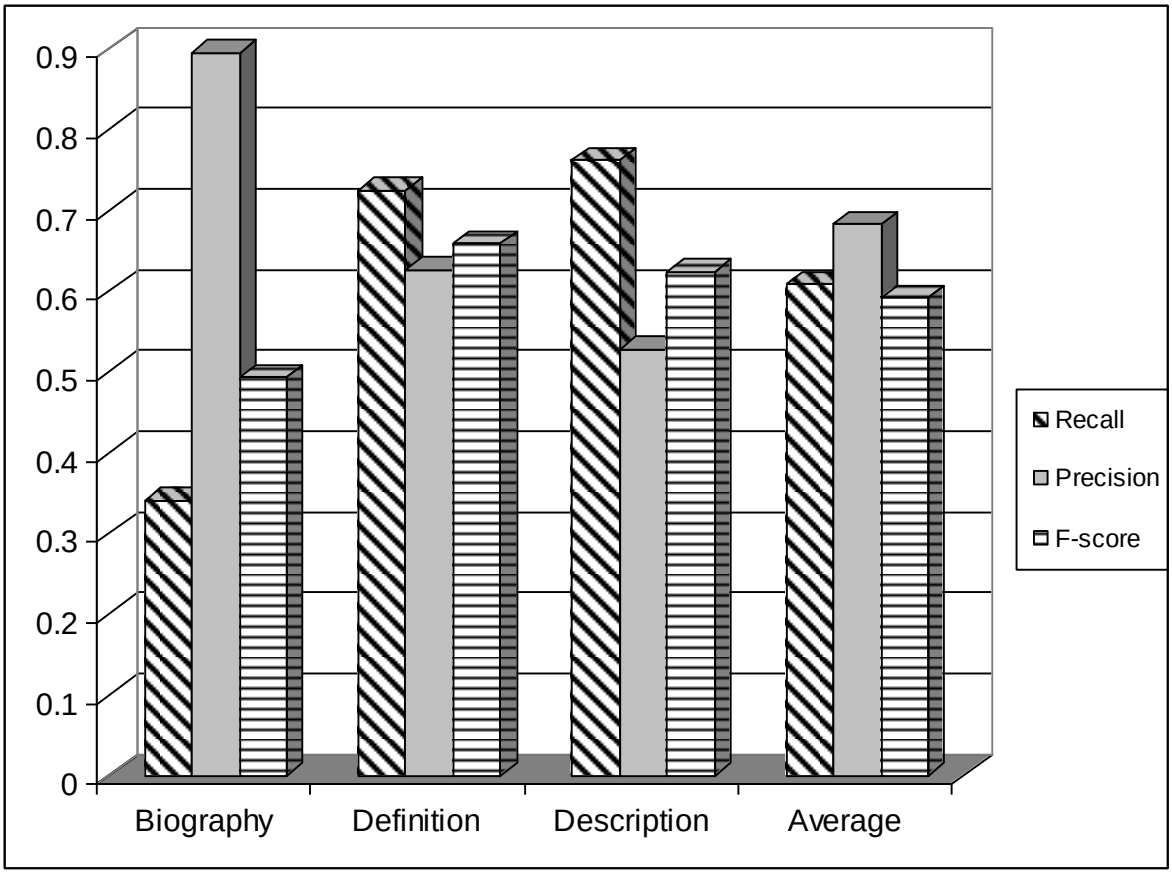

Figure 5.3*: Answer Extraction performance chart*

As shown in Figure 5.3, the average recall, precision, and F-score of the answer extraction component are 0.609, 0.683, and 0.592 respectively. That is, the answer extraction component is able to recall about 60% of the sentences that constitute the correct answer with 68.3% precision. Figures 5.4-5.6 show correct answer, partially wrong answer (i.e., answers that contain unrelated concepts with the question), and no answer examples respectively for different questions.

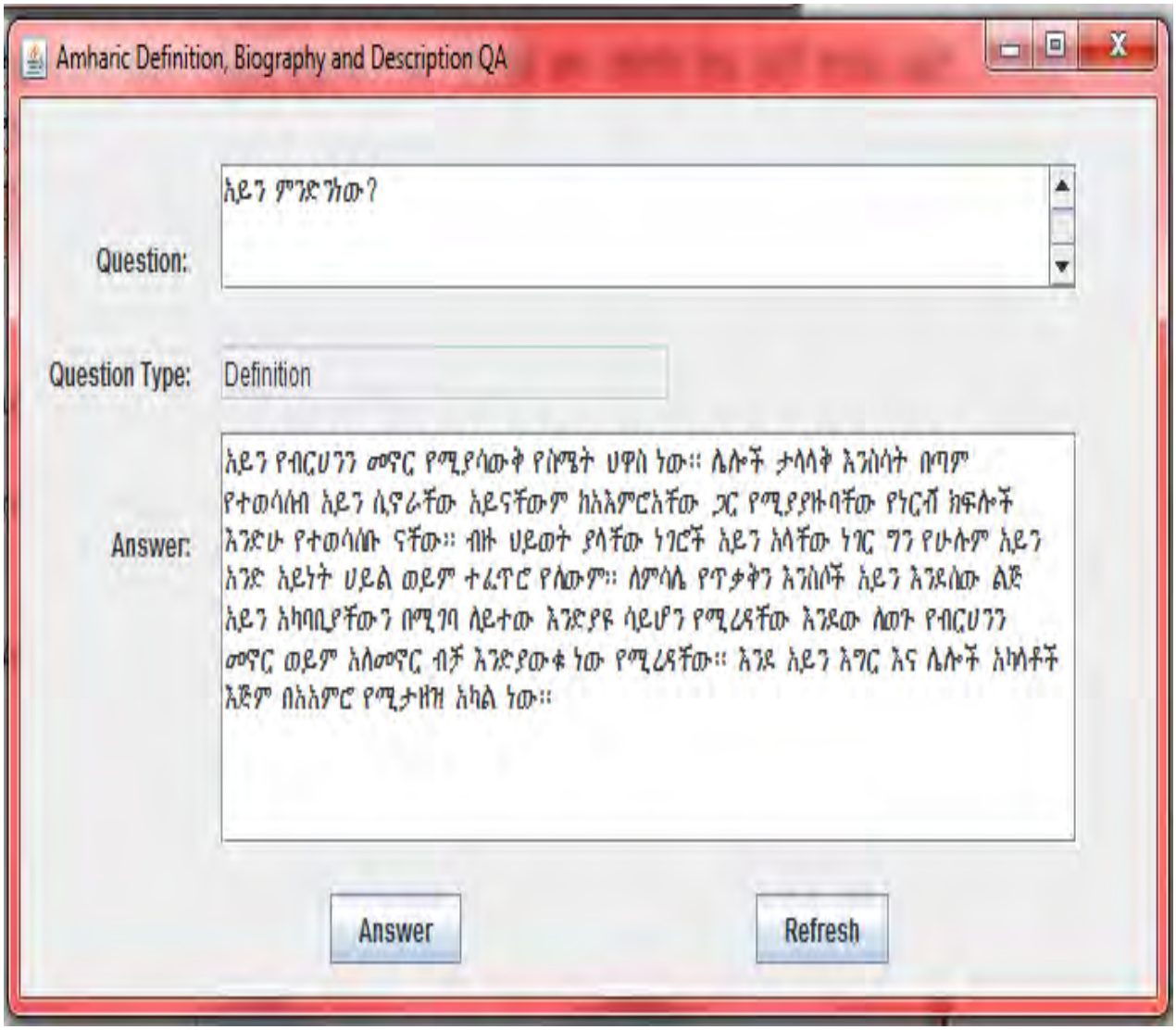

***Figure 5.4**: Screenshot of Correct Answer Example*

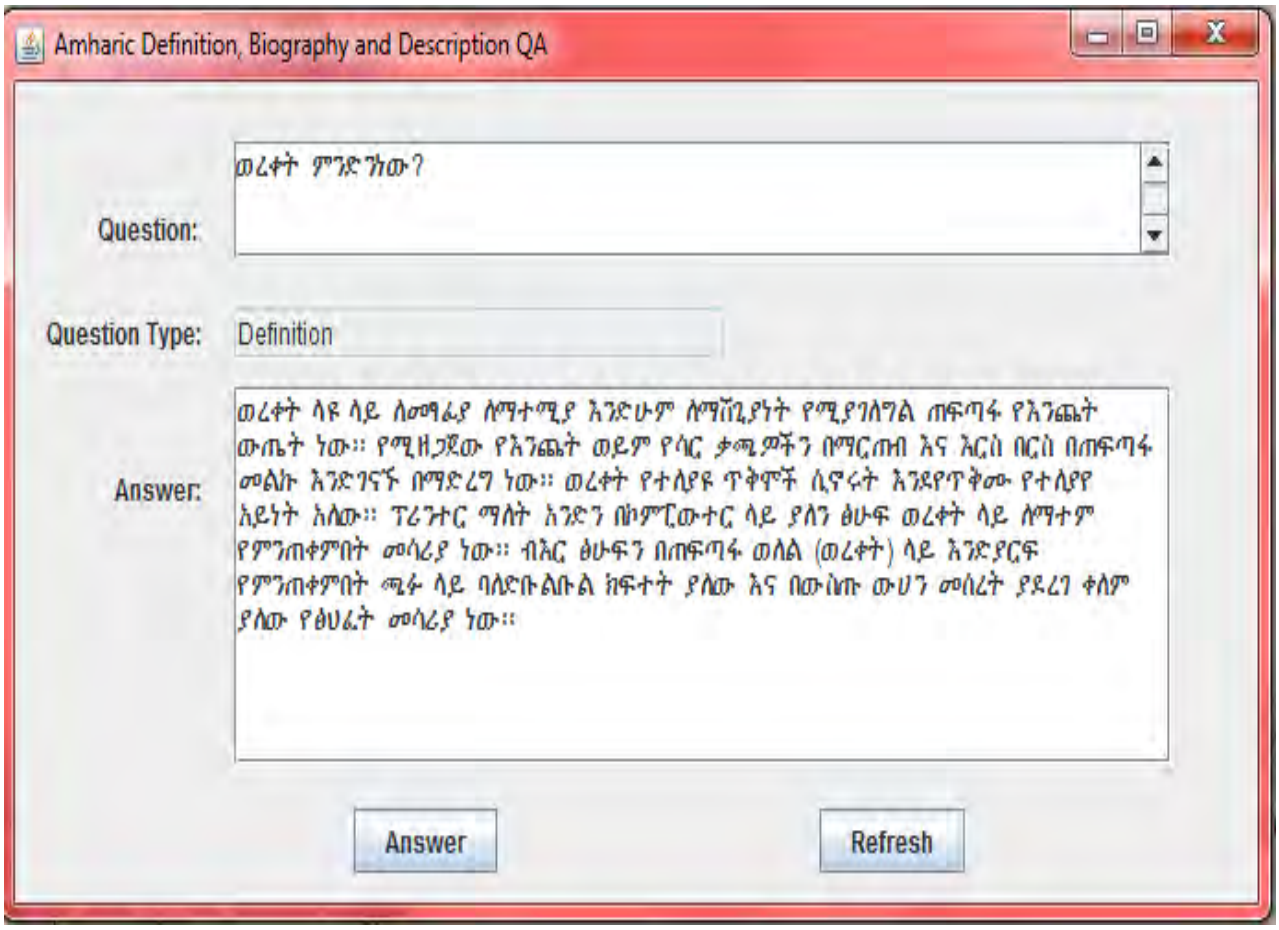

Figure 5.5*: Screenshot of Partially Correct Answer Example*

As shown in Figure 5.5, the last two sentences in the answer for the question **"ወረቀት ምንድን ነው?**" ("what is paper?") should not be included in the answer. Because they are defining the terms "printer" and "pen". Thus this kind of problem would be alleviated by performing detail semantic analysis.

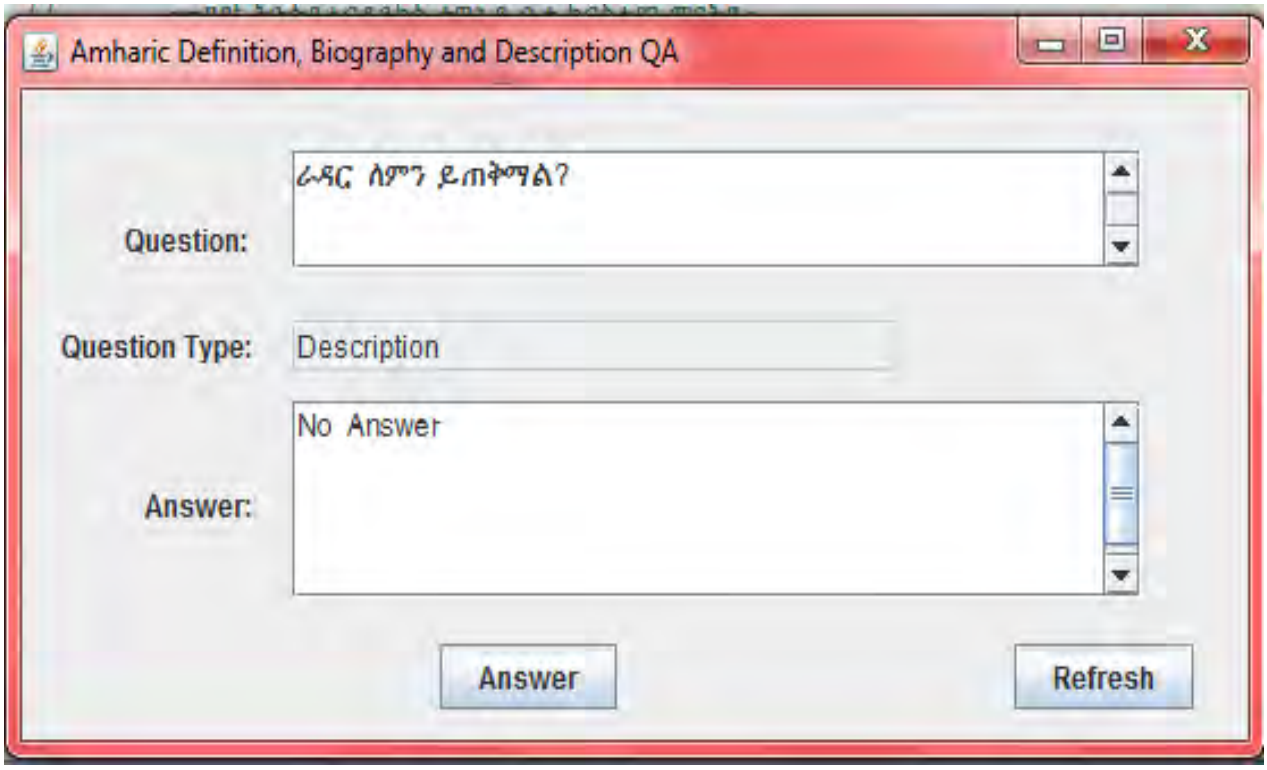

***Figure 5.6**: Screenshot of Wrong Answer Example*

As shown in Figure 5.6, the answer for the question is ―NoAnswer". Even if one of the documents in the corpus contains the text "**...ከዚህ በተጨማሪ የራዲዮ ሞገዶች ነገሮችን "ለማየትም" ሲያገለግሉ ይታያሉ። ራዳር ከኒህ ወገን ሲሆን የሚሰራውም የራዲዮ ሞገዶችን ርቀው ወዳሉ ነገሮች በመላክ ነጥረው ሲመለሱ የሚወስዱትን ሰዓት በመለካት ሩቅ ያሉ ነገሮችን "ለማየት" ይረዳል። በተረፈ ራዳር መሬት ውስጥ ተቀብሮ ያለን ቤንዚን መርምሮ ለማግኘት፣ የአፈርን ኬሚካላዊ ባህርይ ለማወቅ ይረዳል።...**" that discusses the use of the term ―**ራዳር**", the description answer extraction patterns could not match with "**ራዳር ከኒህ ወገን ሲሆን የሚሰራውም የራዲዮ ሞገዶችን ርቀው ወዳሉ ነገሮች በመላክ ነጥረው ሲመለሱ የሚወስዱትን ሰዓት በመለካት ሩቅ ያሉ ነገሮችን "ለማየት" ይረዳል።**" and "**በተረፈ ራዳር መሬት ውስጥ ተቀብሮ ያለን ቤንዚን መርምሮ ለማግኘት፣ የአፈርን ኬሚካላዊ ባህርይ ለማወቅ ይረዳል።**" sentences. As a result the answer for the question becomes ―No Answer".

5.6 Discussion

The proposed question-answering system in this thesis is evaluated from the question classification, document retrieval, and answer extraction aspects. The question classifier is evaluated by computing the percentage of correctly identified question types from the test questions. The evaluation criteria for the other two components were recall, precision, and F-score.

The two methods used in question classification were evaluated by 10 fold cross validation technique in which both have good performance. Minimum percentage values were obtained on definition and description questions. The ambiguity that it faces while classifying these questions is due to the question term "ምንድን ነው", i.e., this interrogative term is used in both question types. Its performance could be maximized by having more training questions. Since the interrogative terms are totally different from definition and description questions, both classifiers have no problem on classifying biography questions. In general, on average we get 98.33% and 83.33% accuracy for the rule based and SVM based classifier respectively.

The other component is the document retrieval. Its performance was evaluated by using two data sets. One of the data sets is analyzed by a stemmer and the other one is analyzed by a morphological analyzer. That is, in this experiment, we aim to compare the effect of stem and root of words in document retrieval. We found that the document retrieval has better performance on the data set analyzed by the morphological analyzer than the stemmed data set.

The answer extraction component is evaluated using 120 questions. It was able to answer questions with 0.609 recall, 0.683 precision, and 0.592 F-score. Since there is no any evaluation benchmark for Amharic QA we could not be able to grade it. Moreover, the QA systems in [3, 32] are factoid and hence we did not compare it with them.

During our evaluation we noted the following points:

- The lack of standard Amharic QA evaluation benchmarks to formally judge the quality of our QA system.
- As was previously mentioned, only those sentences which are actually identified by the answer extraction patterns are processed by the answer extraction component and as the system make no attempt to perform co-reference resolution and so any information which can only be found by relating a number of sentences is lost.

- Even though the techniques introduced in this thesis have performed well, it is clear that there are questions which are not answered correctly and got answers that contain sentences unrelated to them. Improvements to the underlying tools, such as the stemmer, morphological analyzer, name entity recognizer, and summarizer would probably result in improvement of performance.

Chapter Six: Conclusion and Future Work

People need precise information to successfully operate in their environment, support their decisions and solve problems. A gap between an individual‘s background knowledge and information needed in a certain context motivates a search for information. However, there is huge amount of electronic information produced by different individuals, organizations, and countries in the world. There are also large amount of Amharic electronic documents. Though getting precise information is not simple, as a result information retrieval (IR) systems/search engines became essential in everyday life as well as in carrying out professional tasks.

The urge to get as concise information as possible for natural language questions using computers leads to Question Answering (QA). In contrast with traditional document retrieval systems/search engines, which return ranked lists of potentially relevant documents that users must then manually browse through, question answering systems attempt to directly provide users one or more concise answers in the form of paragraphs, sentences or phrases to natural language questions.

6.1 Conclusion

Research on Amharic Question Answering focuses on factoid questions, whose answers are typically named entities such as dates, locations, proper nouns, other short noun phrases, or short sentences. But there are situations, in which someone could want to get the definition and the description of a term, and the biography of a person or an organization. Thus in this thesis we developed an Amharic QA system for biography, definition, and description questions. The QA system consists of four major modules which are document preprocessing, question analysis, document analysis, and answer extraction.

In Amharic one word can be expressed by different characters (—ilel”). This is due to the characters that have the same sound but different structural appearance. Also words could be written in short form. In addition answer source documents and questions could contain stop words that could not contribute for answer extraction. Moreover, a word could have different forms due to the addition of prefixes, infixes, and postfixes.

Thus to address such and other language related issues the question and the documents in the corpus are normalized by the document preprocessing component. The document preprocessing component performs character normalization, short word expansion, stop word removal, stemming, and lemmatization. Except the morphological analyzer used for the lemmatization, the document preprocessing component that we used is similar to that used in [3, 32].

The question analysis component identifies the semantic type of the question by the question classification subcomponent, generates a query from the given question by the query generator, and finally expands the generated queries by the query expander subcomponent. Rule based algorithm and SVM based algorithm are used for the questions classification. Then using the generated queries the document analysis component retrieves relevant documents and filters them using the document retrieval and document filtering components respectively.

For the answer extraction we propose a type-by-type approach. That is, the definition-description answer extraction component extracts sentences/snippets using manually crafted definition and description answer extraction patterns, computes the score of the extracted sentences accordingly, selects non-redundant top ranked sentences using answer selection algorithm, and orders them to keep their coherence. On the other hand the biography answer extraction component generates an answer using a summarizer.

The performances of the question classification, document retrieval, and answer extraction components are evaluated using percentage, precision, recall, and F-score as evaluation metrics. The two techniques utilized in the question classification are evaluated by the 10 fold cross validation technique using 1500 questions (i.e., 150 questions are used for testing on each iteration). The performance of the SVM based question type classifier is 83.3% and that of the rule based question type classifier is 98.3%. That is, the SVM based classifier on average is able to identify the question type of 125 questions correctly out of the 150 test questions and the rule based classifier on average is able to identify the question type of 147 questions correctly out of the 150 test questions. The document retrieval component is tested on two data sets that contain stemmed and lemmatized documents. The F-score on the stemmed documents is 0.729 and on the lemmatized documents is 0.764. The answer extraction component is evaluated by 120 test questions and 300 documents. The average recall, precision, and F-score of the answer extraction

component are 0.609, 0.683, and 0.592 respectively. In general, the algorithms and the tools used have shown promising performance.

6.2 Contribution of the work

The main contributions of this thesis work are outlined as follows:

- The study has adopted other languages non-factoid QA systems techniques to Amharic biography, definition, and description QA.
- The study has implemented a machine learning based and rule based automatic question classification for biography, definition, and description questions.
- The study has identified important natural language processing tasks that have impact on the performance of non-factoid Amharic QA.
- The study showed how a morphological analyzer could enhance the document retrieval performance than a stemmer.
- The study has shown how to identify candidate sentences/snippets for definition and description by using a set of lexical patterns, filter redundant sentences/snippets, select the appropriate ones by ranking them using a statistical approach, and order them to keep the coherence of the definition/description. In addition it also showed Amharic summarizers could be used in non-factoid questions.
- This study also paved a way how to determine a generated answer for a biography by a summarizer is actually a biography or not.

6.3 Recommendations and Future Works

In this thesis we designed an Amharic QA that tries to answer biography, definition, and description question. However, in this attempt we have seen that answering such questions is not simple. Especially to have an outstanding performance such questions demands a number of NLP tools even more than the factoid questions. Thus in this section additional features that can be added to increase the performance of the proposed QA system and future research directions are outlined.

- Since there is no standard Amharic QA evaluation benchmark to formally judge the quality of Amharic QA system, we could not be able to grade our QA system. This is because there is no corpus of Amharic documents and Amharic questions for QA system evaluation. Therefore, it is recommended that such a corpus be prepared for future researches on QA.
- As was previously mentioned only those sentences which are actually identified by the answer extraction patterns are processed by the answer extraction component and as the system makes no attempt to perform co-reference resolution and so any information which can only be found by relating a number of sentences is lost. Developing a system that could perform co-reference resolution is also another issue for research.
- Evaluation results of the research have shown that our QA system is very sensitive to stemming and morphological analysis and hence, effort should be made towards improving the current Amharic stemmer and morphological analyzer.
- The lexical patterns used for answer extraction are crafted manually. This task is so tedious, it requires detailed language knowledge, and might not contain all patterns that are used for definition and description answer extraction. Thus generating patterns using machine learning algorithms would alleviate those problems.
- While extracting snippets/sentences for definition and description questions we used lexical pattern, i.e., texts that match with patterns will be extracted. In some questions this results in the appearance of sentences that are not related to the query. Therefore, development of a semantic analyzer can be one research direction to enhance the QA system‘s performance.
- We used a single document summarizer to generate an answer for biography questions by merging the documents that contain the target to one document. Thus enhancing the performance of the single document summarizer is one research direction; on the other hand developing multi document summarizer is also another research direction.
- Since the answer is generated from different documents keeping its coherence is very important in maximizing the quality of the answer. Hence, different methods such as anaphora resolution, lexical chains, discourse structure, etc. have vital roles in generating coherent answer. Thus developing a tool that can do those tasks is also another research direction.

- This research work has shown that with minimal NLP tools it possible to answer biography, definition, and description question types. Extending this work to other question types like why, list, and how questions is another research direction.
- To improve the performance of a QA system and to support other Amharic based natural language processing applications developing an Amharic part of speech tagger and named entity recognizer is a research direction.
- While writing a question if a user commits a spelling error, the system will blindly start processing the user request without checking the correctness of the words in the question. As a result this could be a source for a wrong answer. Thus developing an Amharic spelling checker would help to avoid this source of error.
- It would be easy for systems like ours to better understand the users‘ intentions and return to them with as much possible answers. Therefore, incorporating an Amharic WordNet will maximize our QA system‘s performance.

References

[1] Indurkhya N. and Damereau F.J., (Eds). Handbook of Natural Language Processing. *2nd Ed., Chapman & Hall/CRC, Boca Raton,* 2010.

[2] Silvia, Q. Advanced Techniques for Personalized, Interactive question Answering on the Web. *In Workshop on Knowledge and Reasoning for Answering Questions, Manchester, pp. 33–40 (2008).*

[3] Seid Muhe. ―TETEYEQ: Amharic Question Answering System for Factoid Questions", *Unpublished MSc Thesis, Addis Ababa University, 2009.*

[4] Lita, L. and Carbonell, J. Instance-based question answering: A data driven approach. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP- 04), pp 396–403, 2004.*

[5] E. Brill, J. Lin, M. Banko, S. Dumais, and A. Ng. Data-intensive question answering. *In Text REtrieval Conference (TREC), 2001.*

[6] Abdessamad Echihabi and Daniel Marcu. A noisy-channel approach to question answering. *In the Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, pp.16-23, July 2003, Sapporo, Japan.*

[7] D. Moldovan, S. Harabagiu, R. Girju, P.Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, and O. Bolohan. LCC tools for question answering. *In Text REtrieval Conference (TREC), 2002.*

[8] Niu, Y. ―Analysis of Semantic Classes toward Non-Factoid Question Answering", *Unpublished PhD Thesis, Department of Computer Science, University of Toronto, Canada, 2007.*

[9] Vanessa Lopez. ―PowerAqua: Open Question Answering on the Semantic Web", U*npublished PhD thesis, Department of Computer Science, The Open University, England, 2011.*

[10] H. Q. Hu. ―A Study on Question Answering System Using Integrated Retrieval Method", U*npublished Ph.D. Thesis, The University of Tokushima, Tokushima, 2006.*

[11] Mengqiu Wang. ―A Survey of Answer Extraction Techniques in Factoid Question Answering", *School of Computer Science, Carnegie Mellon University, USA*, *2006.*

[12] Christopher D. Manning, Prabhakar Raghavan, Hinrich Schtze. An Introduction to Information Retrieval, *Cambridge University Press, New York, NY, 2008.*

[13] Erik Liljenback. ―CONTEXTQA: Experiments In Interactive Restricted Domain Question Answering". *MSC in Computer Science thesis, San Diego University, unpublished 2007.*

[14] Miriam Fernández Sánchez. ―Semantically Enhanced Information Retrieval: An ontology-based approach", *Doctoral dissertation, Unitversidad de Autónoma, Madrid, unpublished 2009.*

[15] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. Modern Information Retrieval. *ACM Press/ Addison-Wesley, Boston, MA, 1999.*

[16] http://en.wikipedia.org/wiki/Tilahun_Gessesse/, *last accessed on February 27, 2013.*

[17] Sasha Blair-Goldensohn, Kathleen R. McKeown, and Andrew Hazen Schlaikjer. A hybrid Approach for Answering Definitional Questions. *In Proceedings of the 12th Text REtreival Conference. NIST, Gathersburg, MD 2003 pp. 185–192.*

[18] Wesley Hildebrandt, Boris Katz, and Jimmy Lin. Answering Definition Questions Using Multiple Knowledge Sources, *In Proceedings of the 12th Text REtrieval Conference (TREC 2003).*

[19] Raud Soricut and Eric Brill. Automatic Question Answering Using the Web: Beyond the Factoid, *Journal of Information Retrieval - Special Issue on Web Information Retrieval, 9(2):191-206, Mar 2006.*

[20] Omar Trigui, Lamia Hadrich Belguith, and Paolo Rosso. DefArabicQA: Arabic Definition Question Answering System, *In Workshop on Language Resources and Human Language Technologies for Semitic Languages, 7th LREC, Valletta, Malta, 2010.*

[21] Masaki Murata, Sachiyo Tsukawaki, Toshiyuki Kanamaru, Qing Ma, and Hitoshi Isahara: Non-Factoid Japanese Question Answering through Passage Retrieval that is Weighted Based on Types of Answers. *In the preceedings of the third IJCNLP, Jan 2008.*

[22] Tatsunori Mori, Mitsuru Sato, Madoka Ishioroshi, Yugo Nishikawa, Shigenori Nakano, and Kei Kimura. A Monolithic Approach and a Type-by-Type Approach for Non-Factoid Question-answering. *In Proceedings of NTCIR-6 QAC Workshop, May 2007, Tokyo, Japan.*

[23] Zhushuo Zhang, Yaqian Zhou, Xuanjing Huang, and Lide Wu. Answering Definition Questions Using Web KnowledgeBases. *R. Dale et al. (Eds.) IJCNLP (2005) LNAI 3651, pp. 498 – 506. Springer, Heidelberg (2005).*

[24] Kian Wei Kor. Improving Answer Precision and Recall of List Questions. *Master of Science, School of Informatics, University of Edinburgh, unpublished 2005.*

[25] Otis Gospodnetic and Erik Hatcher. ―Lucene in Action," *Manning Publications Co., 209 Bruce Park Avenue, Greenwich, 2005.*

[26] Mark Andrew Greenwood. ―Open-Domain Question Answering", *Unpublished Doctoral Dissertation, Department of Computer Science University of Sheffield, UK, September 2005.*

[27] Liddy, E. D. Encyclopedia of Library and Information Science, *2nd Ed. Marcel Decker, Inc. 2003.*

[28] Joachims T. Text categorization with support vector machines: Learning with many relevant features. *In European Conference on Machine Learing, ECML-98, Berlin, Germany, pp. 137–142, 1998.*

[29] Cortes, C. and V. N. Vapnik. Support vector networks. *Machine Learning, 20:273–297, 1995.*

[30] V. Vapnik. The Nature of Statistical Learning Theory, *Springer-Verlag, New York, 1995.*

[31] Steve Gunn. Support Vector Machines for Classification and Regression, *ISIS Technical Report, Image, Speech and Intelligent Systems Group, University of Southampton, 14 May, 1998.*

[32] Desalegn Abebaw. ―LETEQ (ልጠየቅ)-A Web Based Amharic Question Answering System for Factoid Questions Using Machine Learning Approach", *Unpublished MSc Thesis, Addis Ababa University, 2013.*

[33] Dell Zhang and Wee Sun Lee. Question classification using Support Vector Machines, *Department of Computer Science, School of Computing, National University of Singapore, ACM, Toronto, Canada, July 28-August 1, 2003.*

[34] István Pilászy. Text Categorization and Support Vector Machines, *Department of Measurement and Information Systems, Budapest University of Technology and Economics, 2001.*

[35] http://www.lonweb.org/link-amharic.htm, last accessed on May 5, 2013.

[36] http://jrgraphix.net/research/unicode_blocks.php?block=31, last accessed on May 5, 2013.

[37] **ባየ ይማም ፤ የአማርኛ ሰዋሰው ፤ ት.መ.ማ.ማ.ድ.፣**1987.

[38] **ጌታሁን አማረ ፤ የአማርኛ ሰዋሰው በቀላል አቀራረብ ፣** 1989.

[39] Melese Tamiru. ―Automatic Amharic Text Summarization Using Latent Semantic Analysis", *Unpublished MSc Thesis, Addis Ababa University, 2009.*

[40] Apache Lucene - Index File Formats, http://lucene.apache.org/java/2_4_0/fileformats.html, *last Accessed on April 20, 2013*.

[41] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval, *Information Processing and Management: an International Journal, Vol. 24, No. 5, pp. 513-523, Pergamon Press plc, Great Britain, 1988.*

[42] Voorhees, E. M. and D. M. Tice. Building a question answering test collection. *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece, 2000, pp. 200–207.*

[43] Hirschman, L., M. Light, E. Breck, and J. Burger. Deep read: A reading comprehension system. *In 37th Annual Meeting of the Association for Computational Linguistics (ACL-99), New Brunswick, NJ, 1999, pp. 325–332.*

[44] Charniak, E., Y. Altun, R. Braz, B. Garrett, M. Kosmala, T. Moscovich, L. Pang, C. Pyo, Y. Sun, W. Wy, Z. Yang, S. Zeller, and L. Zorn. Reading Comprehension Programs in a Statistical-Language-Processing Class. *In ANLP/NAACL Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems, Seattle, WA, May 2000, pp. 1–5.*

[45] Breck, E., J. Burger, L. Ferro, L. Hirschman, D. House, M. Light, and I. Mani. How to evaluate your question answering system every day and still get real work done. *In Proceedings of the Second International Conference on Language Resources and Evaluation. LREC-2000, Athens, Greece, 2000.*

[46] Jurasfsky D. and Martine J. H. Speech and Language Processing: An introduction to speech recognition, natural language processing, and computational linguistics, 2006.

[47] Getasew Tsedalu, ―Information Extraction Model From Amharic News Texts", *Unpublished MSc Thesis, Addis Ababa University, November, 2010.*

[48] Van Rijsbergen, C.J. Information Retrieval, 2nd edn., *Butterworths, London, U.K., 1979.*

[49] Tessema Mindaye. ―Design and implementation of Amharic Search Engine", *Unpublished MSc Thesis, Addis Ababa University,* 2007.

[50] Michael Gasser. HornMorpho: a system for morphological processing of Amharic, Oromo, and Tigrinya, *Conference on Human Language Technology for Development, Alexandria, Egypt, 2011.*

[51] E. Nyberg, T.Mitamura, J. Callan, J. Carbonell, R. Frederking, K. Collins-Thompson, L. Hiyakumoto, Y. Huang, C. Huttenhower, S. Judy, J. Ko, A. Kupsc, L. V. Lita, V. Pedro, D. Svoboda, and B. V. Durme. The javelin question-answering system at trec 2003: A multi strategy approach with dynamic planning. *In Text REtrievalConference (TREC)*, *2003*.

[52] A. Ittycheriah, M. Franz, and S. Roukos. Ibm's statistical question answering system - trec-10. *In Text REtrieval Conference (TREC)*, *2001*.

[53] H.P. Luhn, ―The Automatic Creation of Literature Abstracts", *In IBM Journal of Research and Development, pp. 159-165, 1958*.

[54] Eduard Hovy, ―Text Summarization", *In The Oxford Handbook of Computational Linguistics, pp. 583-598*, *2005*.

[55] Karel Ježek and Josef Steinberger, ―Automatic Text Summarization (The state of the art 2007 and new challenges)", *In Proceedings of Znalosti 2008, pp. 1–12*, *2008*.

[56] Yihong Gong and Xin Liu, ―Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis", *In Proceedings of ACM SIGIR, pp. 19-25*, *2001*.

Appendix A: Short words and their expansions

ሆ/ል ሆስፒታል

መ/ቅ መጸሐፍ ቅዱስ

መ/ቤት መሥሪያ ቤት

ሚ/ሩ ሚኒስትሩ

ም/ ምክትል

ም/ቤት ምክር ቤት

ሠ/ፌዴሬሽን ሠራተኛ ፌዴሬሽን

ሻ/ ሻምበል

ቤ/መ ቤተመንግስት

ቤ/ክ ቤተክርስትያን

ተ/ ተክለ

ኃ/ ኃይለ

አ/አ አዲስ አባባ

ኮ/ል ኮለኔል

ወ/ ወልደ

ወ/ሪት ወይዘሪት

ወ/ሮ ወይዘሮ

ዓ/ም ዓመተ ምህረት

ዓ/ዓ ዓመተ ዓለም

ዶ/ር ዶክተር

ጄ/ል ጄኔራል

ገ/ ገብረ

ጠ/ሚ ጠቅላይ ሚኒስትር

ጽ/ቤት ጽህፈት ቤት

ፍ/ቤት ፍርድ ቤት

ፕ/ር ፕሮፌሰር

ፕ/ት ፕሬዚዳንት

ት/ቤት ትምህርት ቤት

Appendix B: Stop words list

ሁሉ	ብዛት	ነው
ሁሉም	ብዙ	ነይ
ኗላ	ቦታ	ነገር
ሁኔታ	በርካታ	ነገሮች
ሆነ	በሰሞኑ	ናት
ሆኑ	በታች	ናቸው
ሆኖም	በኗላ	አሁን
ሁል	በኩል	አለ
ሁሉንም	በውስጥ	አስታወቀ
ላይ	በጣም	አስታውቀዋል
ሌላ	ብቻ	አስታውሰዋል
ሌሎች	በተለይ	እስካሁን
ልዩ	በተመለከተ	አሳሰበ
መሆኑ	በተመሳሳይ	አሳስበዋል
መካከል	የተለያየ	አስፈላጊ
የሚገኙ	የተለያዩ	አስገነዘቡ
የሚገኝ	ተባለ	አስገንዝበዋል
ማድረግ	ተገለጸ	አብራርተዋል
ማን	ተገልጿል	አበራርተው
ማንም	ተጨማሪ	አስረድተዋል
ሰሞኑን	ተከናውኗል	እስከ
ሲሉ	ችግር	እባክህ
ስለ	ታች	እባክሽ
ቢቢሲ	ትናንት	እባክዎ
ቢሆን	ነበረች	አንድ
ብለዋል	ነበሩ	አንጻር
ብቻ	ነበረ	እስኪደርስ

እንኳ
እስከ
እዚሁ
እና
እንደ
እንደገለጹት
እንደተገለጸው
እንደተናገሩት
እንደአስረዱት
እንደገና
ወቅት
እንዲሁም
እንጂ
እዚህ
እዚያ
እያንዳንዱ
እያንዳንዳቸው
እያንዳንዷ
ከ
ከኋላ
ከላይ
ከመካከል
ከሰሞኑ
ከታች
ከውስጥ
ከጋራ
ከፊት
ወዘተ
ወይም
ወደ
ዋና
ወደፊት
ውስጥ
ውጪ
ያለ
ያሉ
ይገባል
የኋላ
የሰሞኑ
የታች
የውስጥ
የጋራ
ያ
ይታወሳል
ይህ
ደግሞ
ድረስ
ጋራ
ግን
ገልጿል
ገልጸዋል
ግዜ
ጥቂት
ፊት
ደግሞ
ዛሬ
ጋር
ተናግረዋል
የገለጹት
ይገልጻል
ሲሉ
ብለዋል
ስለሆነ
አቶ
ሆኖም
መግለጹን
አመልክተዋል
ይናገራሉ

Appendix C: Sample test questions and their question type for the rule based and SVM based classifiers

Question Type	No	Question	Identified Question Type by the SVM Based Classifier	Identified Question Type by the Rule Based Classifier
Biography Questions	1	ገብረክርስቶስ ደስታ ማነው?	Biography	Biography
	2	አዲስ አበባ ዩኒቨርሲቲ ማነው?	Biography	Biography
	3	አፍሪካ ማናት?	Biography	Biography
	4	አለቃ ገብረሃና ገብረማሪያም ማናቸው?	Biography	Biography
	5	አለቃ ኪዳነወልድ ክፍሌ ማናቸው?	Biography	Biography
	6	ደራሲ በዓሉ ግርማ ማነው?	Biography	Biography
	7	ብርሃንና ሰላም ማተሚያ ቤት ማነው?	Biography	Biography
	8	ብላቴን ጌታ ኅሩይ ወልደሥላሴ ማናቸው?	Biography	Biography
	9	ዳኛቸው ወርቁ በዛብህ ማነው?	Biography	Biography
	10	ማሞ ውድነህ ማነው?	Biography	Biography
	11	መንግስቱ ለማ ማነው?	Biography	Biography
	12	ደራሲ ጳውሎስ ኞኞ ማነው?	Biography	Biography
	13	ስብሐት ገብረ እግዚአብሔር ማነው?	Biography	Biography
	14	አርቲስት ጥላሁን ገሰሰ ማነው?	Biography	Biography
	15	የኢትዮጵያ አየር መንገድ ማነው?	Biography	Biography
	16	ግብጽ ማናት?	Biography	Biography
	17	የኢትዮጵያ ኦርቶዶክስ ተዋህዶ ቤተ ክርስቲያን ማናት?	Biography	Biography
	18	የጥንታዊ ኢትዮጵያ ጀግኖች ማህበር ማነው?	Biography	Biography
	19	የኢትዮጵያ ፌዴራላዊ ዲሞክራሲያዊ ሪፐብሊክ ማናት?	Biography	Biography
	20	አጼ ሀርቤ ማናቸው?	Biography	Biography
	21	ፊታውራሪ ሀብተጊዮርጊስ ዲነግዴ ማናቸው?	Biography	Biography
	22	አቶ ሀብተማርያም አጐናፍር ማናቸው?	Biography	Biography
	23	ደጃዝማች ሀብተማሪያም ገ/እግዚአብሄር ማናቸው?	Biography	Biography
	24	ደጃዝማች ሀብተሚካኤል ይናዱ?	Unknown	Unknown
	25	ቀዳማዊ ዓፄ ኃይለ ሥላሴ ማናቸው?	Biography	Biography
	26	ዳግማዊ ዓፄ ምኒልክ ማናቸው?	Biography	Biography
	27	ጸጋዬ ገብረ መድህን ማነው?	Biography	Biography
	28	ግርማዊት ንግስተ ነገስታት ዘውዲቱ ማናቸው?	Biography	Biography
	29	የተባበሩት መንግስታት ድርጅት ማነው?	Biography	Biography
	30	ደራሲ ዶ/ር ሀዲስ ዓለማየሁ ማናቸው?	Biography	Biography
	31	ኢንስትክተር መንግስቱ ወርቁ ማናቸው?	Biography	Biography
	32	ወ/ሮ አበበች ጎበና ማናት?	Biography	Biography
	33	አባ ደከር በዛብህ ማናቸው?	Biography	Biography

	34	እቴጌ ማርያም ሥና ማናቸው?	Biography	Biography
	35	ከያኒ ኢዩኤል ዮሀንስ ማነው?	Biography	Biography
	36	ቀሲስ ከፍያለው መራሄ ማናቸው?	Biography	Biography
	37	መስፍን ሀብተማርያም ማነው?	Biography	Biography
	38	አቡነ ተክለሃይማኖት ማናቸው?	Biography	Biography
	39	ዶ/ር ወልደ መስቀል ኮስትሬ ማናቸው?	Biography	Biography
	40	መለሰ ዜናዊ ማነው?	Biography	Biography
	41	አትሌት ጥሩነሽ ዲባባ ማናት?	Biography	Biography
	42	ሻምበል አበበ ቢቂላ ማነው?	Biography	Biography
	43	አዲስ አበባ ማናት?	Biography	Biography
	44	ፕሮፌሰር አሸናፊ ከበደ ማነው?	Biography	Biography
	45	ሙላቱ አስታጥቄ ማነው?	Biography	Biography
	46	ተስፋ ኀህ ኢትዮጵያ ማነው?	Biography	Biography
	47	አትሌት መሰረት ደፋር ማናት?	Biography	Biography
	48	የኢትዮጵያ ቴሌቪዥንና ሬድዮ ድርጅት ማነው?	Biography	Biography
	49	ጤና ቀበና ማህበር ማነው?	Biography	Biography
	50	ኬር ኢትዮጵያ ማነው?	Biography	Biography
Definition Questions	1	ኒሻን ምን ማለት ነው?	Definition	Definition
	2	ሰምና ወርቅ ምን ማለት ነው?	Definition	Definition
	3	ሰረገላ ምን ማለት ነው?	Definition	Definition
	4	ጥናትና ምርምር ምን ማለት ነው?	Definition	Definition
	5	እርሻ ምን ማለት ነው?	Definition	Definition
	6	ኢንቨስትመንት ምን ማለት ነው?	Definition	Definition
	7	ጾም ምንድንነው?*	Unknown	Definition
	8	ትንሳኤ ምንድንነው?	Unknown	Definition
	9	አቡጊዳ ምን ማለት ነው?	Definition	Definition
	10	አልጎሪዝም ማለት ምን ማለት ነው?	Definition	Definition
	11	ስዕል ምን ማለት ነው?	Definition	Definition
	12	ቴያትር ትርጉሙ ምንድንነው?	Definition	Definition
	13	አቶም ምንድንነው?	Unknown	Definition
	14	ባንክ ምንድንነው?	Definition	Definition
	15	አዕምሮ ምንድንነው?	Definition	Definition
	16	ንቃተ ህሊና ማለት ምን ማለት ነው?	Definition	Definition
	17	ክሮማቶግራፊ ምን ማለት ነው?	Definition	Definition
	18	ኮምፕዩተር ምን ማለት ነው?	Definition	Definition
	19	ሥራ ምንድንነው?	Unknown	Definition
	20	መሬት ምንድንናት?	Definition	Definition
	21	ኮረንቲ ማለት ምን ማለት ነው?	Definition	Definition
	22	ኤሌክትሮ መግነጢስ ምንድንነው?	Unknown	Definition
	23	አይን ምንድንነው?	Definition	Definition
	24	ሰንደቅ ዓላማ ምንድንነው?	Unknown	Definition
	25	ጂዎሜትሪ ትርጉሙ ምንድንነው?	Definition	Definition
	26	ግዕዝ ምንድንነው?	Unknown	Definition
	27	የሰዋሰው ፍቺ ምንድንነው?	Unknown	Definition

* Rows are shaded when the two methods differ in question classification.

	28	የመሬት ስበት ማለት ምን ማለት ነው?	Definition	Definition
	29	ፀጉር ምንድንነው?	Definition	Definition
	30	እጅ ምንድንነው?	Definition	Definition
	31	መሠረተ ልማት ምን ማለት ነው?	Definition	Definition
	32	ኢንተርኔት ማለት ምን ማለት ነው?	Definition	Definition
	33	እስልምና ማለት ምን ማለት ነው?	Definition	Definition
	34	ጃቫ ማለት ምን ማለት ነው?	Definition	Definition
	35	ብርቱካን ምንድንነው?	Definition	Definition
	36	የመሬት ከባቢ አየር ምን ማለት ነው?	Definition	Definition
	37	ብርሃን ማለት ምን ማለት ነው?	Definition	Definition
	38	ፀሀይ ምንድንናት?	Definition	Definition
	39	ቴሌቪዥን ማለት ምን ማለት ነው?	Definition	Definition
	40	ጨረቃ ምንድንናት?	Definition	Definition
	41	ትምህርተ ሂሳብ ማለት ምን ማለት ነው?	Definition	Definition
	42	ምህንድስና ምን ማለት ነው?	Definition	Definition
	43	ኔትወርክ ምንድንነው?	Definition	Definition
	44	ሚልኪ ዌይ ማለት ምን ማለት ነው?	Definition	Definition
	45	ራዲዮ ማለት ምን ማለት ነው?	Definition	Definition
	46	ገንዘብ ምንድንነው?	Definition	Definition
	47	ንጥረ-ነገር ምንድንነው?	Unknown	Definition
	48	እሳተ ገሞራ ምንድንነው?	Unknown	Definition
	49	ሙዚቃ ማለት ምን ማለት ነው?	Definition	Definition
	50	ካልሲ ምንድንነው?	Unknown	Definition
Description Questions	1	ስዕል ጥቅሙ ምንድንነው?	Unknown	Definition
	2	ባንክ ምን ጥቅም አለው?	Unknown	Description
	3	የዋስትና ድርጅቶች ምን ተግባር አላቸው?	Description	Description
	4	ንቃተ ህሊና ለምን ይጠቅማል?	Description	Description
	5	ክሮማቶግራፊ ለምን ይጠቅማል?	Description	Description
	6	ኮምፕዩተር ለምን ይጠቅማል?	Description	Description
	7	ሁለት ነጥብ ለምን ይጠቅማል?	Description	Description
	8	ነጠላ ሰረዝ ምን ጥቅም አለው?	Description	Description
	9	ድርብ ሰረዝ ምን ጥቅም አለው?	Description	Description
	10	ሦስት ነጥብ ምን ጥቅም አለው?	Description	Description
	11	ትእምርተ ጥቅስ ምን ጥቅም አለው?	Description	Description
	12	ትእምርተ አንክሮ ጥቅሙ ምንድንነው?	Unknown	Definition
	13	አራት ነጥብ ጥቅሙ ምንድንነው?	Unknown	Definition
	14	አንድ ነጥብ ጥቅሙ ምንድንነው?	Unknown	Definition
	15	ሥርዓተ ነጥቦች ለምን ይጠቅማሉ?	Description	Description
	16	ሥራ ምን ጥቅም አለው?	Description	Description
	17	መሬት ምን ጥቅም አላት?	Description	Description
	18	አይን ለምን ይጠቅማል?	Description	Description
	19	ሰንደቅ ዓላማ ለምን ይጠቅማል?	Description	Description
	20	ጂዎሜትሪ ምን ጥቅም አለው?	Description	Description
	21	ግዕዝ ጥቅሙ ምንድንነው?	Unknown	Definition
	22	እጆች ምን ጥቅም አላቸው?	Description	Description

	23	መሥረተ ልማት ለምን ይጠቅማል?	Description	Description
	24	የኢንተርኔት ጥቅም ምንድንነው?	Description	Description
	25	ብርሃን ምን ጥቅም አለው?	Description	Description
	26	ገንዘብ ምን ጥቅም አለው?	Description	Description
	27	ኤምፒ3 ለምን ይጠቅማል?	Description	Description
	28	ሙዚቃ ለምን ይጠቅማል?	Description	Description
	29	የሲስተም አሰሪ ለምን ይጠቅማል?	Description	Description
	30	ወረቀት ለምን ይጠቅማል?	Description	Description
	31	ስልክ ምን ጥቅም አለው?	Description	Description
	32	ፕሪንተር ለምን ይጠቅማል?	Description	Description
	33	ራዳር ለምን ይጠቅማል?	Description	Description
	34	የራዲዮ ሞገዶች ጥቅም ምንድንነው?	Description	Description
	35	ራዲዮ ምን ጥቅም አለው?	Unknown	Description
	36	ፀሀይ ምን ጥቅም አላት?	Description	Description
	37	ጥርስ ለምን ይጠቅማል?	Description	Description
	38	ቴሌቪዥን ለምን ይጠቅማል?	Description	Description
	39	ወይን ምን ጥቅም አለው?	Unknown	Description
	40	እንጨት ለምን ይጠቅማል?	Description	Description
	41	ኤክሴል ምን ጥቅም አለው?	Description	Description
	42	ፈተና ለምን ይጠቅማል?	Description	Description
	43	ኑሮ በዘዴ መማር ለምን ይጠቅማል?	Description	Description
	44	ቤት ሲሰራ መስኮት አገልግሎቱ ምንድንነው?	Unknown	Definition
	45	ሰዓት ለምን ይጠቅማል?	Description	Description
	46	ኮምፒዩተር ውስጥ የማዘር ቦርድ ጥቅም ምንድንነው?	Unknown	Description
	47	የኤልክትሪክ ማከፋፈያ ተግባር ምንድንነው?	Description	Description
	48	ስታብላይዘር ለምን ይጠቅማል?	Description	Description
	49	ሰላማዊ ሰልፍ ምን ጥቅም አለው?	Description	Description
	50	የፖለቲካ ፓርቲዎች ምን ይጠቅማሉ?	Unknown	Unknown

Declaration

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

_____

Tilahun Abedissa

This thesis has been submitted for examination with my approval as an advisor.

_____

Mulugeta Libsie (PhD)

Addis Ababa, Ethiopia

November, 2013