ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
FACULITY OF INFORMATICS
DEPARTMENT OF COMPUTER SCIENCE

TETEYEQ (ተጠየቅ): AMHARIC QUESTION ANSWERING SYSTEM FOR FACTOID QUESTIONS

By: **Seid Muhie Yimam**

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

June, 2009

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUSTE STUDIES
FACULITY OF INFORMATICS
DEPARTMENT OF COMPUTER SCIENCE

TETEYEQ (ተጠየቅ): AMHARIC QUESTION ANSWERING SYSTEM FOR FACTOID QUESTIONS

By: Seid Muhie Yimam

ADVISOR:

Mulugeta Libsie (PhD)

APPROVED BY

EXAMINING BOARD:

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

Dedication

ለጋሽዬ፦ May God Bless your soul, you drive me the way I don't know, but appreciate it.

ለአለምዬ፦ Aren't you my "Father" who battled to educate her son, I never forget those harsh times you face.

ለሀኒ፦ Do you think I can finish my thesis on time without your love, appreciation and continuous encouragement!

Acknowledgements

All the praises and thanks be to God, the Lord of the Alamîn. Then, there are a number of people who helped me in developing my thesis work. **Dr. Mulugeta Libsie**, my advisor, I thank you for your encouragement starting from my thesis title selection to the end of this thesis work. You showed me how researches will be produced, how the best advisor interacts with his student, and you are really a model for academic professionals. Thank you all the time. **Nico Schlaefer**, the developer of **OpenEphyra Question Answering System**, you are the one who told me the difficult task to be done easily. Hadn't you sent that mail, I had had ended up with a different mind. Thank you very much. **Mark A. Greenwood,** your PhD. Dissertation on **Open-Domain Question Answering** was very helpful. I read it trice and it helped me a lot and thank you. **Tessema**, I have played a lot with your **ሐበሻ search engine** java codes. They helped a lot for my thesis, and I tell you the codes are written very readable. The **Lucene mailing Lists**; hadn't you all be around there, tell me when will I understand the very internals of Lucene API. I appreciate the group for the prompt response, and I will continue to be the member day in and day out. **HU**, thanks for the money you paid. It was a lot for me. **Tamesol Communications**, thank you very much for the question and answer corpus. You are real collaborator while most organizations in Ethiopia don't possess it. **Mulugeta** and **Elias**, I am lucky to have classmates of such boys. You showed me how Java is friendly and more professional. Thank you very much for your uninterrupted helps. **Henok**, you are my always **"የበረታው ትዴ".** Thank you and **Mele** being friends who are alongside to "boost" my working mood. **Hussien J**, thanks for the print. **Rawdi,** thanks for those books and your sisterly encouragement. **Bere** and **Emma**, you were the best seniors in guiding juniors. All **questionnaire respondent**, you are thanked for preparing the "Exam" to test my system. **Honey**, you were newlywed, but it was you who "get-up-and-gos" me to finish on time. Thank you for your lovely coffee and I hope I will make the "Payback" in double! But, **Mam**, words can't count how much you were to me. From the remote countryside KEDIJO, I know the struggling you did to send me school. I am lucky to have a mother like you from such a place. I love you **Mam**.

TABLE OF CONTENTS **PAGE**

LIST OF TABLES **PAGE**

LIST OF FIGURES **PAGE**

Acronyms & Abbreviations

AQA	Amharic Question Answering
API	Application Programming Interface
ASQA	Academic Sinica Question Answering
DARPA	Defense Advanced Research Projects Agency
DLT	Document and Linguistic Technology
EAT	Expected Answer Type
GATE	General Architecture for Text Engineering
IE	Information Extraction
IR	Information Retrieval
NE	Named Entity
NER	Named Entity Recogniser
NIST	National Institute of Standards and Technology
NLP	Natural Language Processing
POS	Part of speech
QA	Question Answering
QARAB	Question Answering for ARABic
RDBMS	Relational Data Base Management System
SEFT	Search for Text IR
TB	Terra Byte
TIPSTER	a program of research and development in the areas of information retrieval, extraction, and summarization funded by DARPA
TREC	Text Retrieval Conference
US	United States
XML	eXtensible Mark-up Language

ABSTRACT

Amharic documents on the Web are increasing as many newspaper publishers started their services electronically. People were relying on IR systems to satisfy their information needs but it has been criticized for lack of delivering “readymade” information to the user, so that the QA systems emerge as best solution to get the required information to the user with the help of information extraction techniques. QA systems in other languages have been extensively researched and have shown reasonable outcomes, while it is the first work for Amharic. Amharic is a less-resourced language and developing a QA system was not done before. A number of techniques and approaches were used in developing the Amharic QA system. The language specific issues in Amharic are extensively studied and hence, document normalization was found very crucial for the performance of our QA system. Experiment has showed that documents normalized bear higher performance than the un-normalized ones. A distinct technique was used to determine the question types, possible question focuses, and expected answer types as well as to generate proper IR query, based on our language specific issue investigations. An approach in document retrieval focused on retrieving three types of documents (Sentence, paragraph, and file). The file based document retrieval is found more important than the other two techniques, i.e., taking the advantages of concept distribution over sentences and less populous answer particles found in a file based retrieval techniques. An algorithm has been developed for sentence/paragraph re-ranking and answer selection. The named entity (gazetteer) and pattern based answer pinpointing algorithms developed help locating possible answer particles in a document. The evaluation of our system, being the first Amharic QA system, has shown promising performance. The rule based question classification module classified about 89% of the question correctly. The document retrieval component showed greater coverage of relevant document retrieval (97%) while the sentence based retrieval has the least (93%) which contributed to the better recall of our system. The gazetteer based answer selection using a paragraph answer selection technique answers 72% of the questions correctly which can be considered as promising. The file based answer selection technique exhibits better recall (0.909) which indicates that most relevant documents which are thought to have the correct answer are returned. The pattern based answer selection technique has better accuracy for person names using paragraph based answer selection technique while the sentence based answer selection technique has outperformed in numeric and date question types. In general, our algorithms and tools have shown good performance compared with high-resourced language QA systems such as English.

KEYWORDS: Amharic Question Answering, Answer Selection Techniques, Sentence/paragraph Re-ranking, Question Answering Evaluation

CHAPTER ONE

INTRODUCTION

1.1 GENERAL BACKGROUND

Huge amount of information is now available in machine-readable form. In 2003, approximately 8 TB of books are assumed to be published per year [16]. It has been also assumed that reading of new scientific material that is produced every 24 hours will take a human being about five years to finish. The number of Amharic documents on the web increases as many news agencies provide their service electronically. As a result, Information Retrieval and Information Extraction are becoming more important for effectively looking up and making use of these information. The traditional search engines have a shortcoming for the concise and complete retrieval of information. Commonly, search engines return the relevant, even, most of the time irrelevant links or document lists to the search keywords which are excessive that users need more effort to acquire the needed information, may be after reading a number of pages for a longer time [1].

While information retrieval is effective by itself, users these days demand a better tool. First, they want to reduce the time and effort involved in formulating effective queries for search engines (users are required to formulate queries that should maximize document matching, and the search engine processes the query as submitted), and secondly they want their results to be real answers—not the list of relevant links. They want to spend less time searching appropriate answers from the lists and more time thinking about what they found and using it for whatever purpose they started the search in the first place. They want a Question Answering (QA) system that is more efficient than the usual search engine, but at the same time a flexible, robust, and not fussy, just like Google. Question answering aims to develop techniques that can go beyond the retrieval of relevant documents in order to return exact answers to natural language questions.

In a traditional document retrieval system, the task of extracting the answer from the retrieved documents falls straightforwardly upon the user, and it becomes a significant analysis burden on the user. QA technology aims to reduce this burden by following document retrieval with a series of advanced processing steps to locate and return the correct answer [14].

These days, QA technology has extensively been researched in different languages. Automatic question answering has become an interesting research area and resulted in a substantial improvement in its performance [2]. The aim of QA is to retrieve exact information from a large collection of documents, such as the Web. The main initiative behind QA system development is that users in

general prefer to have a single (or couple of) answer(s) for their questions rather than having a number of documents to be read like the case of search engine's return [3]. Having a number of documents such as the World Wide Web or a local collection, a QA system should be able to retrieve answers to questions introduced in natural language. "*QA is regarded as requiring more complex natural language processing (NLP) techniques than other types of information retrieval such as document retrieval, and it is sometimes regarded as the next step beyond search engines"* [4, 17].

In the case of search engines, the search statement will be broken down into keywords so that the engine returns links to all of the texts. For example, for the search "*who was the first Chinese in space?"*, the search engine returns links containing texts such as **Chinese, space** and **first** and it will take a considerable amount of time for the user to get the appropriate answer. Whereas, in the case of QA information retrieval, the system returns sentences or phrases instead of documents. For the above question, a QA system will return the assured answer as **Yang Lewie is the first Chinese in space** or simply **Yang Lewie** [6].

For a question given in a natural language, the question type and the anticipated answer type should be firstly analyzed (question analysis). 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** [7, 14]. Where, who, when, which, yes/no, true/false, name-of, etc. are kinds of factoid questions. As an example, "what is a university?" is a definition question where as "where is Mt. Ras Dashen located?" is a where question which seeks location [8]. Some questions are 1) closed-domain (where the questions raised are in a specific domain such as in medicine) and 2) open-domain which are questions almost about everything [4].

In general, factoid questions need very brief and short lined answers. Users expect very concise answers for factoid questions. A question "*Who was the first person to climb Everest?* Or *Where is Taj Mahal located?"* are factoid questions which need a brief answer "*Edmund Hillary and his Sherpa guide Tenzing Norgay were **the first** humans to reach Earth's highest point: the summit of Mount **Everest** in the Himalayas. They reached the top at 11:30 am on 29 May 1953*" and "***Taj Mahal** is **located** in Agra, India*" respectively [9].

For all the above types of questions, QA systems have already been developed in different languages such as Chinese [10, 11, 12], English [9, 13] and so on.

Most QA systems comprise of question processing, document retrieval, and answer extraction components. The **question processing** module is responsible in determining the question types, the

expected answer types, question focus, and determines the proper query to be submitted to the document retrieval component. Determining the question type, i.e., about what the question is, can be done with the help of question particles such as **who** and **where** as well as by understanding the semantics of the question. The semantic of the question will be known with the help of the question focus, a word or group of words specifically related with some question types. The expected answer type is directly related to the question type and the question focuses. The ultimate goal of determining expected answer type is to easily extract the correct answer by the answer extraction module. The question processing module also generates a proper query that will help in matching relevant documents that might bear the correct answer.

The **document retrieval** component is responsible in retrieving relevant documents from a collection. It is comparable to IR systems where IR systems such as search engines deliver relevant documents to the user based on the query submitted. It is clear that the document retrieval component is very essential as an irrelevant document results in a wrong or NO answer. The document retrieval component might incorporate paragraph/sentence retrieval depending on the needs and techniques used in the QA systems.

The **answer extraction** module, which is the very core component of QA systems, involves different techniques in extracting the correct answer. This module will apply different algorithms and techniques to correctly determine the exact answer. It exerts maximum effort in selecting the best answer and incorporates a number of techniques.

The performance of question answering system will be evaluated from different angles. The main evaluation criterion is correctness of the returned answer. The correctness of the returned answer has different variations based on different systems. Some QA systems need the answer to be exact such as person name and place name whereas others consider sentence or paragraphs bearing the correct answer is acceptable as an answer. Some QA systems also accept ranked answers so that the system will be evaluated based on the availability of the correct answer among the top n answers while others strictly require only the top answer as acceptable. The second evaluation criterion is the efficiency of the system towards processing time and memory space. Most of the time, QA systems will take considerable amount of processing time and negatively affect the response time.

Chapter One: Introduction

STATEMENT OF THE PROBLEM

Amharic is written with a version of the Ge'ez script known as ፊደል (Fidel) [18]. The Amharic language has its distinct way of grammatical construction, character (fidel) representation and statement formation [19, 22, 24].

The question construction and answering techniques in Amharic are different from English and other languages. In English, questions will be developed, for example, using "***wh***" words such as "who is the prime minister of Ethiopia?" and so on. But this same question will have different structure in Amharic such as a difference in character and word formation as well as grammatical arrangement and type of question particles used. For example, the above question will be translated as (**የኢትዮጵያ ጠቅላይ ሚኒስትር ማን ይባላሉ?**). This question needs a special consideration to exactly return the correct answer, which is very different from English and other languages question answering techniques. To the best of our knowledge, there is no QA system developed for Amharic so far. Hence, the problem that this research work tries to address is how to develop an Amharic Question Answering system

1.2 MOTIVATION

The Text REtrieval Conference (**TREC**), co-sponsored by the National Institute of Standards and Technology (NIST) and the US Department of Defense, was started in 1992 as part of the TIPSTER (a program of research and development in the areas of information retrieval, extraction, and summarization, funded by DARPA and various other Government agencies) Text program. Its purpose was to support research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies, which have QA track amongst the tracks [5].

Question answering is being extensively researched in English and other languages and has shown excellent improvement since the TREC QA track has been launched. The QA systems for English can't help in answering Amharic questions, as it needs different language dependent processing. Just for a try, we have posed the question" **የኢትዮጵያ ጠቅላይ ሚኒስትር ማን ነው**?" for the very known web based QA system known as START. It can't understand the question at all, not even to try to answer the question. Figure 1.1 shows the result.

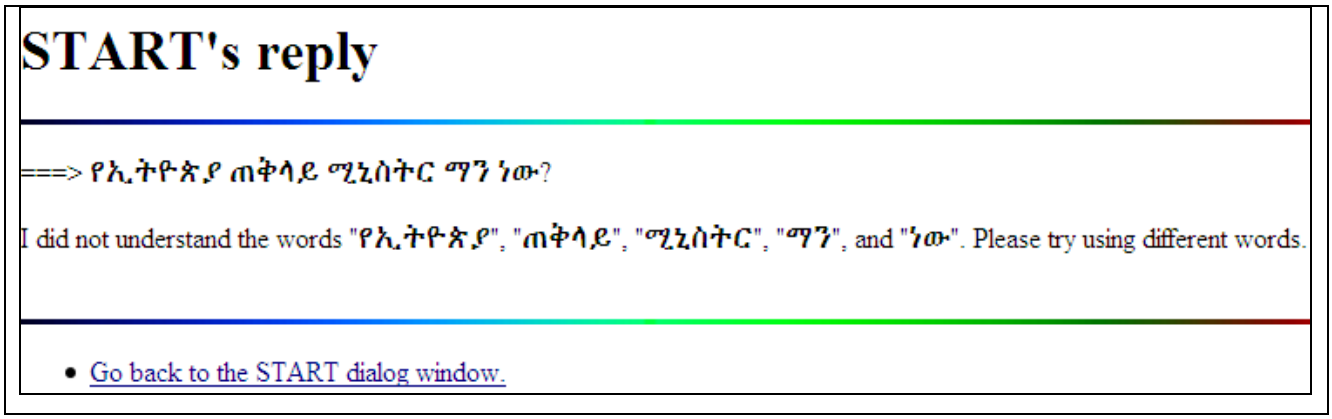

Figure 1.1: An attempt to answer Amharic questions on the START QA system.

Moreover, an Amharic Search engine has been already developed [15] that will help in searching relevant documents of users' request. Unlike a search engine's return, some users are more interested in getting correct answers for their questions. History students or story seekers, Tourists, Online customers, Organization Information Desk users, Service Provider's users such as hospitals prefer an exact answer to their query rather than paged details of a document as of search engines. This specific problem motivated us to study and investigate the possibilities of QA System development for Amharic language.

1.3 OBJECTIVES

GENERAL OBJECTIVE

The general objective of this research work is to develop a prototype for Automatic Amharic Question Answering System for factoid question.

SPECIFIC OBJECTIVES

The specific objectives of this research are:

a. Studying the general grammatical structure of Amharic statements related to factoid question types.
b. Identifying the relationship between Amharic factoid questions and statements.
c. Investigating the different types of question types, expected answer types, question particles
d. Analyzing question and answer patterns.
e. Developing a general architecture of Amharic factoid question Answering.
f. Developing an algorithm for Amharic Factoid Question Answering system.
g. Developing a prototype for the new system.

h. Evaluating the new system.

1.4 SCOPE AND LIMITATIONS

Naturally, Question Answering is a very complex and rigorous task which needs understanding of natural language processing techniques. A full-fledged QA system will require a number of natural language processing tools such as Sentences parser, Chunker, Part of Speech (POS) tagger, Stemmer, Named Entity Recognizer (NER) and so on. Even though some of the NLP tools have been developed by some researchers, they are not publicly available for integrating with our system. Having these limitations in mind, our scope will be:

a. Answering only "ማን", "የት", "መቸ", and "ስንት" type of Amharic questions.
b. Closed-domain factoid questions types, specifically Amharic News collected from different newspapers.
c. Developing Gazetteers (list of known Person name, Place name, …)

Most of the Amharic NLP tools are done as an academic exercise by MSc. students and are not publicly available. Due to the unavailability of some of these NLP tools, we are going to use manual and simple techniques just as a means of demonstration.

1.5 METHODOLOGY

The main methodologies, tools, data sources, and testing strategy we have used for this thesis work are discussed in the following subsections.

LITERATURE REVIEW

For better understanding of QA systems, we have reviewed a number of related works done on QA for different languages such as English, Arabic, Hindi, and Chinese. Open Source Question Answering Systems, **OpenEphyra** and **AnswerFinder** [23], have been studied and a number of QA techniques have been learned. As the research focuses on Amharic language, different language specific features and properties have been studied in light of QA systems.

TOOLS

In researching for Amharic Question Answering (AQA), Java programming language has been employed as a major developmental tool for the prototype. A Lucene searching and indexing API, that has been used in [15] has been modified for the document retrieval module of our system. For the NLP

tools, especially for the Named Entity Recognizer (NER), as there is no such tool easily available to integrate, manual NER (Gazetteers) for person name and place name is employed. For the numerical and date type of questions, we have developed a full-fledged rules (patterns) using regular expressions.

DATA SOURCES

A large number of Amharic corpuses (approximately 15600 news articles) are collected from the Web and Ethiopian newspapers. Besides, nearly 12000 questions have been collected from [53], the Web and different documents for question classification and testing purpose.

QUESTIONNAIRES

In studying and analyzing Question and Answer patterns, besides studying Amharic grammar books, questionnaires have been prepared that were distributed to different people to have better coverage of question particles as well as question and answer patterns. In addition, the questionnaires also helped us to test the performance of the system.

PROTOTYPING

We have built a prototype to test the algorithms and techniques developed.

TESTING

We have made objective testing for our system. Testing is done to check the performance of our system using recall and precision. Both the named entity based (gazetteer based) and the pattern based techniques have been tested.

1.6 APPLICATION OF RESULTS

As QA is an extension to search engines, the AQA system will be employed in retrieving short answers in Amharic for factoid questions quickly, concisely and completely. Besides, it will have a great contribution in Amharic E-learning by providing correct answers to students saving substantial amount of time. Mobiles usually have small memory capacity and limited screen width to read full documents looking for exact answers. For users who want to retrieve Amharic information using mobiles, AQA can be the ultimate solution.

More specifically, the AQA system will be used to provide information about an organization automatically (replacement of the traditional information desk). In a traditional information desk, people usually contact the information desk personnel about details of the organization such as who the manger of that organization is, where the office of a staff is located, who to talk to about a specific

topic (such as payment, employment, and so on). It can also replace the traditional answering machine (automatic telephone answering). With the absence of actual information personnel, AQA will replace the human being and can give accurate and automatic answer to users, a good addition for dialogue system.

1.7 THESIS ORGANIZATION

The rest of the thesis is organized as follows. Chapter 2 discusses the different issues in question answering as a literature review. This Chapter lays the foundation in understanding what a question answering comprises of, what techniques and algorithms will be incorporated, and the tools which are the main backbone of QA systems. Chapter 3 is devoted to discuss related works done on QA systems in different languages. Chapter 4 discusses issues and consequences related to Amharic towards QA.

Many language specific issues such as the writing system have been extensively presented. The architectural and design issues of our system are discussed in Chapter 5. The main components of our system, the functional operation module and the specific sub-component of each module are briefly discussed in this Chapter. Chapter 6 is devoted to discussing the main implementation issues of our new QA system. The algorithms, techniques and methods used in how the system has been successfully developed are discussed in this Chapter. Chapter 7 is devoted to the evaluation of the system and the results as well as the limitations of the system. Chapter 8 concludes the thesis by outlining the benefits obtained from the research work. It also shows some research directions that can be accomplished in QA for Amharic.

CHAPTER TWO

LITERATURE REVIEW

In this Chapter we will concentrate on addressing Question Answering system development strategies. The first section presents the differences and similarities between Information Retrieval (IR) and Information Extraction (IE). The next section will cover details on general QA architectures. The remaining sections will discuss details on QA components, particularly on techniques and approaches in Question Analysis, Document Retrieval, and Answer Extraction components of a QA system. Finally we will discuss the Lucene API including the basic classes used for indexing and searching as well as construction of queries together with the common IR models used by the search engine community.

2.1 INFORMATION RETRIEVAL (IR) AND INFORMATION EXTRACTION (IE)

Information retrieval has been researched extensively mainly to help users in getting relevant documents from large collection of free-text documents. The way IR tackles the problem of document retrieval is based on the closeness of the document and the query submitted to the IR system. IR will not try to present answers to users explicitly. This was the critics of IR so that the need of IE came about. The IE technique involves NLP tools for precisely indicating a correct text. There should be deep analysis of queries (i.e., user questions) to understand the user's intention as well as deep analysis of the document to extract correct answers (sentences or passages). In the case of IR, a simple technique is sufficient to extract content-rich words from the query and applying stemming to make more uniformity of document retrieval that will be applied during indexing too [14].

In information retrieval, queries tend to be more general and lengthy while in IE the queries should be specific and shorter in number of query words. The traditional IR focuses on retrieving related documents and highlighting the excerpts of related documents for the user. Part of the document which contains the query term will be highlighted for the user to give more attention near those highlights. In contrast, the task of question answering, based on IE techniques, is to identify the exact answer or answer bearing passage/sentence for the submitted query (question) [14, 38]. In the case of information retrieval, the system will retrieve related documents and still the user will be involved in analyzing the documents. Whereas, in information extraction, the system will analyze the query and extract the fact so that it will be submitted to the user for readymade usage [14, 40]. GATE (General Architecture for Text Engineering) defines Information Extraction as follows [55]:

> *"Information Extraction is not Information Retrieval: Information Extraction differs from traditional techniques in that it does not recover from a collection a subset of documents which are hopefully relevant to a query, based on key-word searching (perhaps augmented by a thesaurus). Instead, the goal is to extract from the documents (which may be in a variety of languages) salient facts about prespecified types of events, entities or relationships. These facts are then usually entered automatically into a database, which may then be used to analyse the data for trends, to give a natural language summary, or simply to serve for on-line access."*

Information extraction is all about extracting structural factual data mostly from unstructured text (web pages, text documents, office documents, presentations, and so on). IE usually uses data mining tools, NLP tools, lexical resources and semantic constraints for better efficiency. On the other hand, IR is used to retrieve unstructured data that the user latter performs some kind of computation to get the structural text. Manning, *et al*, defines IR as follows [39]:

> *Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).*

Despite these significant differences, IE is not by any means independent and not incomparable to IR. Most IE systems such as the case of QA and Text Summarization involve IR at the front-end and apply its own techniques to extract the required information. The IR component of IE systems will retrieve the relevant documents, actually by receiving IE specific queries, and hand over the resultant document to the IE components for further processing and text or information extraction [14, 23].

2.2 Architecture of Question Answering

A typical pipeline question answering architecture has four components; question analysis, document retrieval, passage (sentence) retrieval and answer extraction [26, 47]. Figure 2.1 shows the pipeline architecture of QA systems [26]. In this architecture, the Question Analyzer is responsible to analyze the question that is determining the proper expected answer type and formulating proper queries for the Document Retriever. The Document Retriever will retrieve the top n related documents that will be subjected to the Passage Retriever latter. The Passage Retriever will extract passages that pinpoint possible answer strings. The final component, Answer Extractor, will extract the correct answer from the ranked extracted passages.

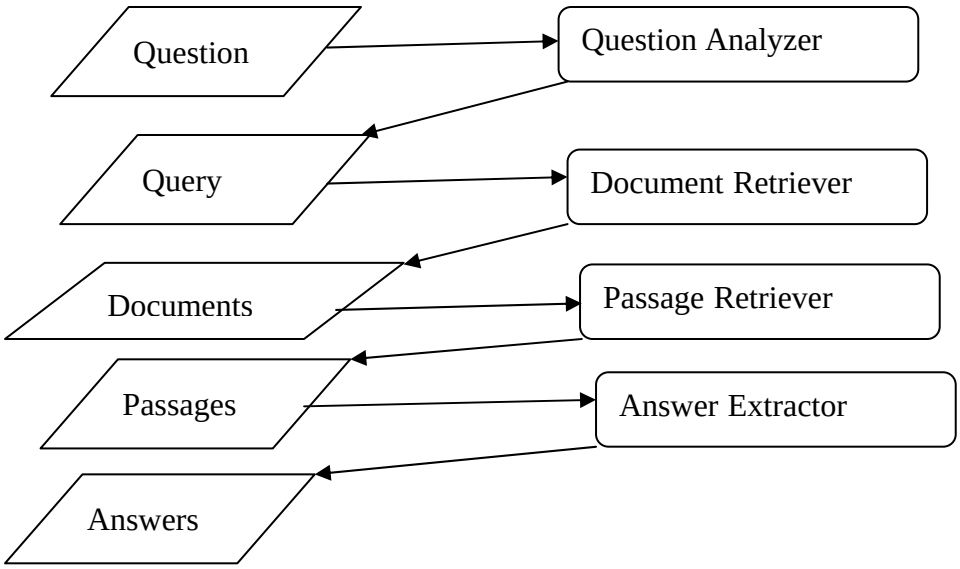

Figure 2.1: A typical pipeline question answering architecture

2.3 QUESTION ANALYSIS

Almost any question answering system will contain a question analysis component. Question Analysis is the most important component of question answering. In the question analysis stage, the type of question will be analyzed. The question type further illustrates what will be the expected answer type. It is the question analysis stage that is also responsible for constructing proper query for the IR component of the QA system. Correctly identifying the expected answer type will help the later stage of answer extraction to correctly identify answers [14, 23]. Therefore, wrong question analysis means that the IR component will retrieve wrong documents as well as the answer extraction component will extract wrong answer or no answer.

Question types can be broadly categorized as LOCATION, PERSON/ENTITY, DEFINITION, NUMERIC, EXPLANATION/LIST, TRUE/FALSE, TIME/EVENT, CHOOSE, and so on.

LOCATION, PERSON and NUMERIC question types are the focus of this study. Based on the question type, an expected answer type will be identified. Therefore, the question analysis component of a QA system will play a great role in determining question types and identifying answer types.

In this subsection, we will first briefly explain the details of question types as well as the expected answer types. Secondly, we will explain the techniques and methods used in identifying answer types

knowing the question type. Finally, we will discuss the query formulation process, which is the last stage of question analysis.

2.3.1 QUESTION TYPOLOGY

There are many ways to ask a question and many ways to answer the same question. Since users may use different versions of a question, and a document may also be represented in different versions, a system should have a capability of combining the different question versions to match with expected answer types [23, 27, 47]. The question type corresponds to the question syntactic form. The detection of a question type gives us a clue to determine the different possible expressions of the answer and will also serve to extract the other question features such as the question focuses. Some researchers [14] use simply the **Wh**-words such as where, when, which, who, and so on to determine the question type. Table 2.1 shows the question types, the measure that is used, and the unit of the measure for numerical questions.

Table 2.1: Measures, Question, Types, and Units

Measures	Question Type	Units
Length	How long, short, tall, height, low, wide, width; what distance	ft, feet, in, inches, yard, mile; mm, millimeter, cm, meter, km, kilometer, light-years.
Time	How long, old, young; what time, what duration.	sec, second, min, minutes, hrs, hours, day, week, month, year, century
Speed	how fast, slow, what speed/velocity	x/y (where x is from Length, y from Time)
Area	how many ‘units’, what area	square x, sq x, where x is from Length; acre, hectare
Volume	how many ‘units’, what volume	cubic x, where x is from Length
Weight	how heavy, many units, what weight	oz, ounce, lb. pound, gm., gram, kg., kilogram
Currency	how much, what cost, price; what prize, salary.	dollars, cents, pound, yen, yuan, franc, lira, mark, shilling, euro, Birr, etc.

2.3.2 ANSWER TYPES

Answer types are used by the QA system as a matching criterion during query submission to the IR system so those candidate answers will be returned for further processing (Answer Extraction) [29].

Answer types are directly related with question types. In addition to the question word, the answer type is also sometimes related with the focus of a noun in the question query [28]. For example for the question "**የኢትዮጵያ ዋና ከተማ ማን ይባላል**?" (What is the capital **city** of Ethiopia), the word **ከተማ (city)** will further indicate that the expected answer type is city name. Questions that will have a question word "**የት ይገኛል** and **የት** (where, which)" will have location name as an answer type. A single answer type may match different forms of question construction. For example, the questions [**የመጀመሪያው ሐብታም የጠፈር ተጓዥ ማን ነው**? (**Who** is the first rich person on space?), **ለመጀመሪያ ጊዜ ጠፈር ላይ የተጓዘው ባለሃብት ስም ማን ይባላል**? (**What is the name of** the rich person to be on space for the first time), **ከሃብታሞች ለመጀመሪያ ጊዜ ጠፈርን የጎበኘው ማን ይባላል**? (Among the riches, **who** happened to visit space for the first time?)] all will have same person name as their answer type. Therefore, an answer type is a relation where all forms of variant questions are mapped to a proper answer type and variants of answer phrasings will be matched to a correct question form [29]. For this purpose, different researchers [14, 23] developed two level categories of answer types as top level (coarse) and fine grained classes for each coarse category. Table 2.2 shows coarse and fine grained answer type categories that will help in classifying questions.

Table 2.2: The answer type hierarchy used for question classification

♥ Amount
✪ Money
✪ Percent
✪ Measurement
✓ Time
How Often
Age
✓ Distance
✓ Speed
✓ Temperature
✓ Area
✓ Mass
✓ Computer
✓ Other
♥ Currency Unit
♥ Reward
✪ Award
✪ Prize
✪ Trophy
✪ Medal
♥ Date
✪ Month
♥ Proper Name
♥ State Motto
♥ State Bird
♥ State Flower
♥ State Tree
♥ State Nickname
♥ Organization
♥ Planet
♥ God
✪ Egyptian
✪ Greek
✪ Roman

♥ Person
✪ Male
✪ Female
♥ Location
✪ City
✪ Province
✪ Country
✪ National Park
✪ Lake
✪ River
✪ Cemetery
✪ Continent
✪ US State
♥ Language
♥ Nationality
♥ National Anthem
♥ Religion
♥ Space Craft
♥ Job Title
♥ Quoted Text
♥ Zodiac Sign
♥ Birthstone
♥ Address
♥ Colour
♥ Colour List
♥ Unknown Proper Name
♥ Chemical Element
✪ Symbol
✪ Name
♥ Chemical Compound
✪ Symbol
✪ Name

2.3.3 DETERMINING EXPECTED ANSWER TYPE

Knowing the expected answer type is a very important part of question analysis. Knowing the expected answer types for unseen questions will greatly improve the performance of a QA system. Unable to identify the correct answer type means that the QA system will return wrong answer or just no answer

by a later component of a QA system. Hence, developing set of answer types will highly maximize the performance of QA systems [14, 23, 47]. Naturally, a single question can be posed in different ways so that the set of answer types would be wide enough to accompany these variations of questions.

For Amharic Language, we have identified hierarchical structure of answer types that will be presented in Chapter 6.

In addition to the expected answer types, identifying a question focus is equally important in identifying the correct answer. Therefore, identification of the focus of a question should be given higher attention. Table 2.3 shows sample questions with question focus and expected answer types.

Table 2.3: Sample Amharic questions, Focus and Expected Answer Type (EAT)

Question	Focus	EAT
በአፍሪካ ትልቁ አገር ማን ነው? (What is the largest country in Africa?)	**አፍሪካ**, **አገር** (Africa: Country)	**አገር** (country)
የሱዳን ጠቅላይ ሚኒስትር ማን ነው? (Who is the prime minister of Sudan?)	**ጠቅላይ ሚኒስትር** (prime minister)	**የሰው ስም** (Person name)
አንድ ኪሎ ስኳር ስንት ብር ነው?(How much does one kilo of sugar cost?)	**ስኳር** (Sugar)	**ዋጋ** (price)
የኢድ አል አድሀ በዓል መቼ ነው?(When is the Eid Al Adha Holiday?)	**ኢድ አል አድሀ** (Eid Al Adha)	**ቀን** (Day)

MANUALLY CONSTRUCTED RULES FOR AUTOMATIC ANSWER TYPE CLASSIFICATION

We can manually construct rules to automatically classify question types. Unlike automatic classification technique which is based on training question and answer sets, this approach will not require more hand labeled training data. The rule will be developed with set of regular expressions that will match same answer type for related questions [48]. This method requires vast number of question set such as [20, 21] analysis to correctly match different related question texts. A single rule can match different question formation as a rule depends on the bag of words and semantics entities of the question [23]. For our research, we have identified rules specifically for person name, place name, and numerical question types to be presented in Chapter 6.

AUTOMATIC QUESTION CLASSIFICATION

The aim of question classification is to determine the type of the answers the user is expecting to receive from a given question. A number of researches have been done on automatically classifying

questions. It is supposed that the automatic classification techniques have main drawbacks such as a need of more training data.

The best method of automatic classification is to employ traditional IR systems for indexing a number of questions and their type so that later it will be possible to determine the type of unseen question type by searching the index [23].

2.3.4 QUERY FORMULATION

The question analysis component of a QA system is responsible in formulating queries that will be submitted to the IR component of the QA system so as to improve the performance of the total system [23].

Most QA systems will assume bag-of-words in the question as a valid query to the traditional IR component which applies stemming and stopwords removals. But, most of the time, some of the words in a question (question particles) are more important in determining the correct document that might contain the answer. For example, in the question "**ኢትዮጵያ ውስጥ ስንት የመንግስት ዩኒቨርሲቲዎች አሉ?**" (how many governmental universities are available in Ethiopia?)" a bag-of-word approach will have only three words (**ኢትዮጵያ** (Ethiopia), **መንግስት** (Government) and **ዩኒቨርሲቲ** (University)) that might help in detecting the correct document where actually **ስንት ነው** (how many) is a very important word removed as stopwords [23, 44, 45, 46].

Query formulation can be done either by expanding the query based on the expected answer type or based on expansion of words in the question. One way of query expansion based on the expected answer type is to create a semantically based index of documents based on the expected answer type. This means, a given document's paragraph, or even sentence will be analyzed semantically so that it will be labeled as **location name, person name, numerical,** etc. as its expected answer type. This technique, while efficient, is difficult to produce since determining the semantic group of a document such as PersonName, Location, Measurement, and so on is tedious as it needs extraordinary analysis. The semantic web will potentially improve this task.

Alternatively, the IR query will be expanded to include some terms which will help in detecting related documents. This approach is well suited for some type of questions such as those that require measurement answers like distance, time, and age [23]. For example, for the question "**የአፍሪቃ ዋንጫ መቸ ይጀመራል?** (When will African Football cup start?)", we can add extra terms such as **1-30,**

2000-2002, መስከረም-ጳጉሜ (September-August) that can be inferred from the interrogative word መቸ (when) so that the IR will include these terms for searching.

The first expected answer expansion technique is not applicable since determining the semantic group of every page/paragraph during indexing is difficult. The second approach cannot be applied for open domain question answering since most search engine IR components have maximum limits of query construction, but it is possible to apply it on closed domain corpuses [23].

The alternative approach, query formulation based upon question words, is to expand the query based on words in the question. It includes expanding queries based on word synonyms. In this approach an English language QA query expansion could be possible to use WordNet† for a word possible synonym expansion [23, 44, 45, 46]. For our research work, we will use manual synonym indexing as an alternative approach since Amharic WordNet is not available. We have obtained list of word synonyms from a website [52].

2.4 DOCUMENT RETRIEVAL

Current question answering systems rely on document retrieval as a means of providing documents which are likely to contain an answer to a user's question. A question answering system heavily depends on the effectiveness of a retrieval system. If a retrieval system fails to find any relevant document for a question, further processing steps to extract an answer will inevitably fail too [23, 25].

Question answering always contains an IR subsystem that will help in identifying documents or passages which may contain an answer for the question [30]. The document retrieval component presents ranked documents so that the answer extraction component will act up on it in a later stage. Some QA systems use the IR subsystem to retrieve related documents where further processing remains to be the task of subsequent components such as passage/sentence extraction system. Some QA systems use the IR subsystem to directly perform the passage/sentence extraction itself.

DOCUMENT RETRIEVAL APPROACHES

There are a number of document retrieval techniques used for IR systems. For our system, we used an open source Lucene IR API. Some of the techniques used in retrieving documents are detailed below.

Stemming: The stemming algorithm is a process for removing the common morphological and inflexional endings from words in a given language. Its main use is as part of a term normalization

† A machine-readable lexical database organized by meanings; developed at Princeton University

process that is usually done when setting up Information Retrieval systems. If stemming is applied to words during indexing, document retrieval will apply the same stemming technique applied during indexing to match related documents. While it is good to include the morphological variants of a term searching, stemming will most of the time make more documents to be returned; possibly making correct answers ranked down the list. A more elegant way of document retrieval could be indexing documents without applying stemming and during retrieval **morphological variants** of the term will be passed so that only the one with the correct match will be returned.

Paragraph retrieval: while document retrieval is sufficient, it will leave the burden of answer bearing passage/sentence pinpointing for the subsequent components. An alternative to document retrieval could be to retrieve passages in the document retrieval component so that the task of the remaining component will be limited to correct answer extraction.

BENEFITS OF QUERY EXPANSION FOR DOCUMENT RETRIEVAL

In section 2.3.4, we have discussed different techniques that will be used in expanding a query. In Chapter 6, we will explore the impacts, if any, of query expansion on document retrieval for AQA. Queries can be expanded either using expected answer type or using question words. In the case of query expanding using expected answer types, some answer pinpointing words will be included to expand the query. For example, for the question "**ባህርዳር ከአዲስ አበባ ምን ያህል ይርቃል?** (How far is Bahr Dar from Addis Ababa?)", distance measurement words such as **ሜትር** (meter), **ኪሎ ሜትር** (kilo meter), etc., can be included with the other words. Besides, queries can be expanded based on word synonym. Hence query expansion helps to match wider document coverage. The evaluation of these techniques is discussed in Chapter 6.

2.5 ANSWER EXTRACTION

In the document retrieval stage, most answer promising documents will be retrieved, while in answer extraction stage, the most possible answer will be extracted [31]. A trivial approach to extract answers from documents is to randomly choose a single word or phrase as an answer. While it is very simple, it is probable that the returned word or phrase would be wrong. A more logical way of extracting answers would be to extract all answer texts and rank them according to their frequency of occurrence in a relevant document. It is a complex way of presenting answers than randomly extracting answers.

In a semantic type answer extraction technique, answer texts with similar expected answer type will be extracted. Two answers can be considered equivalent if they are identical as stated below [23].

> *Two answers are considered equivalent if and only if all the non-stopwords in one answer are present in the other or vice-versa.*

IMPROVED ANSWER RANKING

In searching using the Lucene search API, similar answers having different ranks might be returned so that they will be considered as different answers. Consider an answer excerpts "ጥቅምት 13 2001 (October 13 2001) and 13/2/2001". Both excerpts are identical answers but the baseline search will consider them differently. For this research, we have used different techniques of detecting identical answers in different modes. We will cover the details in Chapter 6.

DYNAMICALLY EXPANDING THE ANSWER TYPE HIERARCHY

The answer type hierarchy defined in the previous section helps in extracting correct answers. In English, in addition to the answer type hierarchies, WordNet can be used to expand answer types that a question seeks. For our research we have to use set of regular expressions that will match expected answer type of a question to extract the correct answer. The answer extraction section of Chapter 6 explains how specifically we develop the regular expression to match expected answer type as an answer from a given sentence/document.

PATTERN BASED ANSWER EXTRACTION

A different method of extracting correct answers is with the help of matching patterns. The patter for the answer will be formulated after analyzing a number of questions with their answers. Person names, place names, dates and times, and other types will have rules that will be defined to match a given question and answers. While pattern based answer extraction has higher precision, it has suffered from limitations like shortage of large pattern libraries to match wider question types and the technique needs a high recall document retrieval technique to retrieve all possible documents containing possible expected answers [58]. If the document retrieval component does not retrieve the possible answer excerpts in a document, the pattern remains unable to answer the question [14, 23]. Constructing question and answer pattern is time consuming and needs a lot of sample question-answer corpuses. Heuristic pattern recognition will be employed in our work to identify possible answers to foreign person names related questions.

NAMED ENTITY RECOGNIZER BASED ANSWER EXTRACTION

Named Entity Recognizer (NER) is a technique that is used to label different entities such as Person Name, Place, Number, dates and times, and so on in a document. NER has been used for information extraction. Current text-based question answering systems usually contain a NER as a core component. The rationale of incorporating a NER in a QA system is that many fact-based answers to questions are entities that can be detected by a NER that will considerably reduce the task of finding an answer. NER basically can be used in the final stage of answer extraction module to filter out sentences that might not contain the expected answer types. Suppose the question type is place and an excerpt contains no expected answer type (i.e., Place), then NER will remove that sentence considering it as irrelevant and will not be considered for answer extraction [31, 45].

A QA system typically uses both taxonomy of expected answers and taxonomy of named entities produced by its NER to identify which named entities are relevant to the question. Hence, the answer extraction component will check if there is a match between these taxonomies to determine the correct answer [45].

2.6 THE LUCENE API

Lucene is a high performance, scalable IR library. It helps to add indexing and searching capabilities to applications. Lucene is a mature, free, open-source project implemented in Java; it is a member of the popular Apache Jakarta family of projects, licensed under the liberal Apache Software License. It has facilities for text indexing and searching that can be integrated in to applications.

We discuss the Lucene API because we will use it for the document retrieval stage of our QA system. Below we will discuss some basic components of Lucene: Indexing, Query Parsing, and Searching [43].

2.6.1 INDEXING

Indexing is the processing of the original data into a highly efficient cross-reference lookup (index) in order to facilitate searching. The index stores statistics about terms in order to make term-based search more efficient. Lucene's index falls into the family of indexes known as an inverted index. An index contains a sequence of documents. A document is a sequence of fields where a field is a named sequence of terms (strings).

Lucene has five basic classes for indexing: IndexWriter, Directory, Analyzer, Document, and Field [43]. Below is a short description of each class.

IndexWriter is the central component of the indexing process. This class creates a new index and adds documents to an existing index. The IndexWriter class can be considered as an object that gives us write access to the index but does not let us read or search it.

The **Directory** class represents the location of a Lucene index. It is an abstract class that allows its subclasses to store the index as they see fit. IndexWriter then uses one of the concrete Directory implementations, **FSDirectory** or **RAMDirectory**, and creates the index in a directory in the file system.

Before text is indexed, it is passed through an **Analyzer**. The Analyzer, specified in the IndexWriter constructor, is in charge of extracting tokens out of text to be indexed and eliminating the rest. If the content to be indexed is not plain text, it should first be converted to it. Analyzer is an abstract class, but Lucene comes with several implementations of it. Some of them deal with skipping *stop words* (frequently used words that don't help distinguish one document from the other, such as *a*, *an*, *the*, *in*, and *on*); some deal with conversion of tokens to lowercase letters, so that searches are not case-sensitive; and so on. Analyzer is an important part of Lucene and can be used for much more than simple input filtering. We have modified and used the AmharicAnalyzer developed by Tessema [15].

A **Document** represents a collection of fields. We can think of it 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. **Fields** of a document represent the document or meta-data associated with that document. The original source (such as a database record, a Word document, a chapter from a book, and so on) of document data is irrelevant to Lucene. The meta-data such as author, title, subject, date modified, and so on, are indexed and stored separately as fields of a document.

Each document in an index contains one or more named fields, embodied in a class called **Field**. Each field corresponds to a piece of data that is either queried against or retrieved from the index during search.

2.6.2 QUERY PARSING

Query parsing is the way in which queries will be parsed that will be appropriate for the searching component of Lucene. Depending on the kind of query, the searching facility will retrieve documents related to the query. We have different kinds of query construction techniques in Lucene such as: TermQuery, RangeQuery, PrefixQuery, BooleanQuery, PhraseQuery, WildcardQuery, and FuzzyQuery which are briefly described in the sequel.

TERMQUERY

The most elementary way to search an index is for a specific term. A term is the smallest indexed piece, consisting of a field name (such as title, modified date, subject, actual file content, and so on) and a text-value pair. TermQuerys are specifically useful to retrieve documents by a keyword. It will be very important in searching databases which are indexed with a unique key such as ISBN of a book or ID number of a student so that an exact match will be retrieved.

RANGEQUERY

Terms are ordered lexicographically within the index, allowing for efficient searching of terms within a range. Lucene's RangeQuery facilitates searches from a starting term through an ending term. The beginning and ending terms may either be included or excluded.

PREFIXQUERY

Searching with a PrefixQuery matches documents containing terms beginning with a specified string.

BOOLEANQUERY

The various query types discussed so far can be combined in complex ways using BooleanQuery. BooleanQuery itself is a container of Boolean *clauses*. A clause is a subquery that can be optional, required, or prohibited. These attributes allow for logical AND, OR, and NOT combinations. A BooleanQuery can be a clause within another BooleanQuery, allowing for sophisticated groupings.

PHRASEQUERY

An index contains positional information of terms. PhraseQuery uses this information to locate documents where terms are within a certain distance of one another. For example, suppose a field contained the phrase "the quick brown fox jumped over the lazy dog". Without knowing the exact phrase, it is possible to find this document by searching for documents with fields having ***quick*** and ***fox*** near each other. Actually a plain TermQuery can do the trick to locate this document knowing either of those words; but in this case we only want documents that have phrases where the words are either exactly side by side (*quick fox*) or have one or more word(s) in between (*quick* [***some irrelevant term(s)***] *fox*). The maximum allowable positional distance between terms to be considered a match is called *slop*. *Distance* is the number of positional moves of terms to reconstruct the phrase in order.

WildcardQuery

WildcardQuery is a somewhat specialized query that allows querying for words with unknown characters. For example: \b*t" => matches both \bet" and \beat", despite the very large semantic gap.

This query type is very expensive in terms of processing time, since a lot of terms in the vocabulary would match a wildcard query, thus a large set of posting lists needs to be retrieved.

FuzzyQuery

FuzzyQuery is a query that provides a way to query for similar matches. The similarity can be based on Levenshtein distance[†]. One example could be “beer” which is similar to “bear”, “beep”.

2.6.3 SEARCHING

Lucene’s searching capability is equally important to its counterpart of indexing. Whatever data indexed in a Lucene index will be meaningless unless it is made convenient for searching. Like the indexing API, the search API, has some core classes that make searching accomplishable. Some of the important searching classes are **IndexSearcher, TermQuery, Term,** and **Hits.**

IndexSearcher is to searching what IndexWriter is to indexing, the central link to the index that exposes several search methods. We can think of IndexSearcher as a class that opens an index in a read-only mode. It offers a number of search methods, some of which are implemented in its abstract parent class **Searcher**; the simplest takes a single **Query** object as a parameter and returns a **Hits** object.

TermQuery is the most basic type of query supported by Lucene, and it is one of the primitive query types. It is used for matching documents that contain fields with specific values.

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.

The **Hits** class is a simple container of pointers to ranked search results; that is documents which match a given query. For performance reasons, Hits instances do not load from the index all documents that match a query, but only a small portion of them at a time.

2.7 IR MODELS

All IR-systems rely on a background model which supports the intention of the IR system and thus provides structures and techniques to support it. A common characteristic for all of these models is that index terms which are words of semantic value are identified from the document collection. Also of importance is that the models consider various terms of different importance, and the interpretation of

[†] http://en.wikipedia.org/wiki/Levenshtein_distance

words somewhat differ [57]. Next we will introduce two very much discussed models in the IR community.

BOOLEAN MODEL

The Boolean model introduced in [57] is an information retrieval model which is based on Boolean algebra. Its concept somewhat lies in the name of the model; it is based on the binary weights of documents. The Boolean model retrieves documents based on a simple prediction to be relevant or non-relevant. A basic search process in an IR-system built on the Boolean model yields a Boolean expression where query terms must be present to represent a match. If none matches are found, then nothing would be retrieved. If a strict match is found, then we have a hit and that particular document would be retrieved. Some decades ago this model was highly popular in the search community, but the drawback of the model was too substantial for it to survive in the field. The major drawback is its foundation; the binary selection of documents based on strict queries. By strict it means queries where all query terms must be present to represent a hit, no partial match. This basics of the model does not yield a good retrieval performance, since documents which semantically would match based on the context would be discarded based on the binary search decision. The binary model remains widely used for databases and strict data retrieval applications. And, this has its root in another drawback; the model is too data oriented, while the semantics are left out.

The Boolean model works very well for databases where relevance and ranking of hits is not a necessity. One example could be searching in a repository for a document with a tagged code, identifier or something like that. For example search for a document containing the unique identifier code “ISBN 0-201-39829-X”. There has been some attempt to enhance the Boolean model, such as the Extended Boolean Model described in [57]. This enhanced Boolean model implements one of the Boolean model’s major drawbacks which is the functionality of partial matching and term weighting. Despite this drawback elimination, the extended Boolean model has not achieved any widespread usage.

VECTOR SPACE MODEL (VSM)

The vector space model is a model based on the realization that the Boolean model is not sufficient in retrieval of documents based on a loose coupling between words and their semantics. The coupling is somewhat non-existing. The VSM model was introduced in 1975 by Salton, and is thus not to be

considered as a state-of-the-art model [57]. The vector space model introduces a new approach which allows partial matching between documents, instead of the regular Boolean matching model which is either a match or no match data retrieval.

The way this is accomplished is that the VSM sees a document as a vector which contains one entry per unique term in the whole document collection (each dimension corresponds to a separate unique term). A query is also to be looked at as a document, hence a vector. Now we have a document collection which is a huge set of vectors representing one document each, and then we have a query which is also to be looked at as a document. A highly common approach to calculate the relevance between a query and a document is to calculate the angle between the two vectors using the dot-product (cosine similarity[†]) function.

Theoretically VSM has a drawback in that the index terms are assumed to be independent of each other, which may imply lack of semantical encapsulation by the model. This drawback is the reason for many of the scalability issues the inverted index suffers from. In VSM each index term is independent, and thus is indexed as a single searchable unit, which means that we end up with huge amount of data.

The other IR models, such as the ***probabilistic model*** and the ***Fuzzy set model*** covered in detail in [57] are alternatives being used in many IR search engines.

[†] Cosine similarity is a measure of similarity between two vectors of n dimensions by finding the cosine of the angle between them.

CHAPTER THREE

RELATED WORK

In this Chapter we will present related works on QA done in other languages. To the best of our knowledge, there is no research work done for Amharic on QA.

3.1 QUESTION ANSWERING FOR ENGLISH

The work in [23] investigates a number of novel techniques for open-domain question answering. Investigated techniques include: manually and automatically constructed question analyzers, document retrieval specifically for question answering, semantic type answer extraction, and answer extraction via automatically acquired surface matching text patterns. Besides Factoid QA techniques, the paper briefly investigated approaches in definitional questions.

The novel techniques in the paper are combined to create two end-to-end question answering systems which allow answers to be found quickly. AnswerFinder[‡] answers factoid questions such as *"When was Mozart born?"*, whilst Varro builds definitions for terms such as "what is *aspirin?"*, "what does *Aaron Copland* mean?*"*, and "define *golden parachute"*. Both systems allow users to find answers to their questions using web documents retrieved by Google. Together, these two systems demonstrate that the techniques developed in the paper can be successfully used to provide quick and effective open-domain question answering.

The work in [44] focuses on techniques used to extract answer strings from a given sentence or paragraph. The main aim of the paper is, given a question Q and a sentence/paragraph SP that is likely to contain the answer to Q, how an answer selection module selects the "exact" answer sub-string A ⊂ SP. The work focuses only on correct answer extraction techniques; but it should be combined with an end-to-end QA system that will accept a question, generate query for document retrieval and determine the expected answer type. One of the techniques employed in extracting the correct answer is to develop an answer pattern where each sentence will be matched to it. The answer string extraction module first selects 100 relevant sentences from the retrieved documents which are ranked based on the relevance of the sentence to the question.

[‡] *AnswerFinder* is a project that is developed by the Centre for Language Technology at Macquarie University

Generally, three techniques of answer extraction were considered: algorithmic which is based on knowledgebase approach processing; pattern matching that are learned from different questions from the web in an unsupervised manner; and statistical approach based on machine translation technique. Here we will discuss the first two techniques since they are helpful for our work.

The knowledge base approach relies on several types of knowledge such as answer typing (Qtargets), semantic relationship matching, paraphrasing, and so on. In the Qtargets technique, first the question is parsed and the TextMap[†] tool determines its answer type (Qtargets) such as PERSON-NAME, PLACE-NAME, and so on. For this purpose, typology has been developed for 185 answer types. For example for the question "How tall is Mt. Everest?", TextMap will determine the Qtarget as DISTANCE-QUANTITY.

The semantic relationship technique is used to select candidate answers, for example in one sentence, based on the semantic relationship of the answers with the remaining words. For instance, if a text "**Jack Ruby, who killed John F. Kennedy assassin Lee Harvey Oswald**" has been retrieved for the question "**Who killed Lee Harvey Oswald**?", the system should answer **Jack Ruby** as a correct answer even though **John F. Kennedy** seems to be the correct answer because of the logical relation of the subject and the verb based on the question meaning.

Paraphrase technique focuses in reformulating the question to possible answer patterns reserving the meaning of the original question because sometimes a sentence with good wording of the question might not help in identifying the exact answer. For example, the question "**Who is the leader of France**?" might have a 100% word overlap with the sentence" **Henri Hadjenberg, who is the leader of France's Jewish community, endorsed confronting the specter of the Vichy past**." which leads to wrong answer of **Henri Hadjenberg** while the sentence "**Bush later met with French President Jacques Chirac**" with 0% word overlap and exact answer of **Jacques Chirac**. So the question can be reformulated as "**who is the leader of French, who is the president of France, who is the president of French"** and so on with synonym expansion.

The pattern based answer selection technique applies matching surface-oriented patterns to pinpoint the answer string. One technique of pattern matching is to reformulate the question to a declarative sentence form (for example **who is the president of US?** will be reformulated as the **president of US is <*complement*>, <complement> is the president of US, etc.**) so that the sentence verbatim with its

[†] It is a search engine for entities: the important people, places, and things in the news. http://www.textmap.com/

answer as a completion will be retrieved. The problem with this technique is that it might return some wrong answers, such as "**The president of US is *perfectly legitimate because he has been elected by the people***". A possible pattern can be hand crafted for every type of question but it can't be guaranteed that it will cover all types of questions properly. The reviewed work applies techniques that will help in learning patterns automatically by submitting some type of Qtargets and answer pairs to the Internet search engines such as Google and Yahoo so that a complete pattern of questions has been developed.

We have reviewed the work of Aunimo [56] that focuses on question typology and feature sets. The paper discussed that the question typology is very important in mapping question types to answer types. Accordingly, it has been proposed that three requirements have been set for good question typology. Firstly, the types of the typology are the answer types (also called targets) for the questions. So that the question "*who is the inventor of television*" has a question type of PERSON where the answer type will be person too. Secondly, the typology should be such that the questions can be automatically classified according to it. That means, there has to be a feature set based on which the classification is possible. Further, the features must be such that they can be inferred programmatically from the questions. Thirdly, the typology should be generalizable, which means that it should apply to different domains and languages.

Hence, 18 question classes have been identified which are used for question classification with the help of 700 question sets. The feature sets, that are terms that further help in classifying questions, have been also identified. The research shows techniques that will help automatically extract features from questions so that it can be matched with identified feature sets to successfully construct the question typology.

3.2 QUESTION ANSWERING FOR ARABIC

QARAB is a QA system developed for the Arabic language [32]. QARAB takes natural language questions expressed in the Arabic language and attempts to provide short answers. In the paper, it has been indicated that QA in Arabic language is difficult since there is slow progress on Arabic natural language processing. Some of the problems identified are:

- Arabic is highly inflectional and derivational, which makes morphological analysis a very complex task.

- The absence of diacritics (which represent most vowels) in the written text creates ambiguity and therefore, complex morphological rules are required to identify the tokens and parse the text.
- The writing direction is from right-to-left and some of the characters change their shapes based on their location in the word.
- Capitalization is not used in Arabic, which makes it hard to identify proper names, acronyms, and abbreviations.

Given a set of questions posed in Arabic, QARAB assumes the following to extract the correct answer.

- The answer exists in a collection of Arabic newspaper text extracted from the Al-Raya newspaper published in Qatar.
- The answer does not span through documents (i.e., all supporting information for the answer lies in one document).
- The answer is a short passage.

The QARAB system has three main components: processing the input question, retrieving the candidate documents (paragraphs) containing answers from the IR system, processing each one of the candidate documents (paragraphs) in the same way as the question is processed and returning sentences that may contain the answer.

The system first collects news documents (the IR system). The collected documents will be indexed and constructed as relational database which stores different relations such as ROOT_TABLE (*Root_ID, Root*) – to store the available distinct roots of the terms extracted from the Al-Raya document collection (one row per root). Secondly the NLP system will be used to determine the part of speech of words (verbs, nouns,…), feature of each word (gender, number,…), and mark proper nouns in the text (personal name, location name, time, …).

In QARAB, documents will be first analyzed, tokenized and stored in RDBMS tables. At the next level of question processing, as there were no solid NLP researches done on Arabic, shallow language understanding technique has been used. For possible answer extraction, bag of words will be submitted and only matched documents will be returned. As QARAB is intended to answer proper name, they have used interrogative particles to determine the expected answer types. After the bag of words are submitted and the type of expected answer is determined, correct answer (such as person name,

location name, …) will be searched from the top ranked documents with the help of type finder, and proper name finder (external systems developed to find person name, location name, …).

3.3 QUESTION ANSWERING FOR CHINESE

Marsha Chinese question answering system [38] focuses on evaluating techniques employed for Chinese language. Marsha consists of three components, query processing, Information retrieval (Hanquery search engine), and the answer extraction module just like many other QA systems. The query processing component will analyze the Chinese questions submitted and formulate a formal query that will be submitted to the IR component. The search engine component will retrieve related candidate documents from the database. The answer extraction module will extract correct answers or candidate answers that will be submitted to the user.

The query processing module carries out the following steps.

1. A question will first be matched to a template so that its type will be determined. Nine question types and 170 templates are prepared to match unseen questions. If a question matches more than one template, such as **how many** and **how many dollars**, it will be matched with the longest template. Table 3.1 shows some of the question types and templates that have been used for this purpose.

Table 3.1: Question types and corresponding templates

Question Type	Templates (Translated from Chinese)
PERSON	which person
LOCATION	which city
ORGANIZATION	what organization
DATE	what date
MONEY	how many dollars
PERCENTAGE	what is the percentage
NUMBER	how many
TIME	what time
OTHER	what is the meaning of

2. Question particles such as **who, how many,** and so on will be removed from the query since they will not help in detecting relevant documents.
3. Named entities will be identified and passed as a word after segmentation.
4. The query is segmented to proper Chinese words.
5. Stopwords will be removed from the query.
6. Finally the query will be formulated for the IR component based on the above steps and submitted to the Hanquery search engine.

The Hanquery search engine [41] retrieves documents related to the question using the query generated by the query processing module and submits the result to the answer extraction module. Finally the answer extraction module extracts the exact answer using techniques such as calculating similarity between query and document, named entity recognition, create pool of candidate answers, and selecting the best answer to return to the user.

The other research work we have reviewed in the Chinese language is the work in [42]. The Documents and Linguistic Technology (DLT) system has components query type identification, query analysis, retrieval query formulation, document retrieval, text file parsing, named entity recognition and answer entity selection. The DLT system has identified 13 categories of query types: **who, what_country, what_city,** and soon**.** The research shows that among these categories, **who, what_city and what_country** match most queries, which is 59.5% (119 query matches out of 200 queries). The query analysis stage of the QA is used to analyze the query and extract from it likely phrases and keywords which will help in retrieving related documents. The query analysis incorporates tokenization (segmentation) and POS tools to appropriately extract phrases which include numbers, quotes, names, verbs, and so on.

In retrieval query formulation stage, phrases identified in the previous stage will be given a weight to indicate how important they are in retrieving the document. After a proper weight is given to phrases, they were then conjoined into a Boolean retrieval query with the phrase of lowest weight first. For document retrieval, Lucene API has been used with a proper language specific tokenizer and stemmer. Specifically a language specific tokenizer and stemmer from the Lucene sandbox (a Lucene repository of third party contributions) has been incorporated. While indexing, they have used sentence-by-sentence techniques considering each sentence as a sentence by looking at appropriate Chinese sentence marker (double-byte punctuation character) which is unique in sentence marking. In order to retrieve documents, the Boolean query composed in the previous stage was submitted to Lucene using

the standard Lucene query language which supports the usual functions such as exact phrases and Boolean operators.

Text file parsing simply extracts the text of a 'document' (i.e., sentence indexed by Lucene) from its XML tags that match the query. The fundamental assumption of the research work is that having a given query type there will be corresponding Named Entity Types. For example, if the query type is **who** then the expected NE type will be **proper_name**. For the 13 identified query types in Chinese, 13 NE types: **proper_name, country**, and so on have been identified. During NE recognition, all instances of NEs of the appropriate types are identified in each candidate document after first segmenting it. The final stage of this QA system is the identification of a particular NE within a document to return it as the answer to the question. This is done by scoring each NE instance using a measure which incorporates the number of co-occurring key phrases, their assigned weights and their distance from the NE.

The QA in [47] is a Chinese question answering system, named as Academia Sinica Question Answering (ASQA), which outputs exact answers to six types of factoid question: **personal names, location names, organization names, artifacts, times,** and **numbers**. ASQA comprises of four components: Question Processing, Passage Retrieval, Answer Extraction, and Answer Ranking.

When ASQA receives a question, it is analyzed by the Question Processing module to obtain question segments, question types, and question focuses. Since Chinese do not have word delimiter, a Chinese segmentation tool CKIP AutoTag has been used to tokenize and part of speech tagging. Six coarse-grained question types (PERSON, LOCATION, ORGANIZATION, ARTIFACT, TIME, and NUMBER) and 62 fine-grained question types have been identified. The question processing module will identify the question type and extracts the question focus (a word or a phrase in a question that represents the answer, and is more informative than the question type). Documents will be preprocessed for proper segmentation and part-of-speech with the CKIP AutoTag tool. Further documents have been segmented into small passages using three punctuation marks ",!?" and indexed by Lucene. The question word segments will create a Lucene query without query expansion to search the index. The Lucene boosting (^) and required (+) operators will be included with the query to give higher significance to the question focuses and type of query terms. Queries are sent to the character-based and word-based indices. The passages derived from the indices are merged after removing duplicate passages. The merged passages are then sorted according to the scores given by Lucene, and the top five are sent to the next module for answer extraction.

A two step answer extraction process has been performed. First, Named Entity Recognition (NER) system is used to retrieve passages and obtain answer candidates. Second, the extracted named entities are filtered based on the expected answer types derived in the question processing phase. They have used a Chinese NER engine, Mencius, to identify both coarse-grained and fine-grained NEs.

After NER processing, the extracted named entities are filtered according to their NE categories to select answer candidates. To do this, a manually constructed mapping table containing information about question types and their corresponding expected answer types has been used. NEs whose types are not found among the expected answer types are removed. The remaining NEs are the answer candidates. In addition to filtering with a mapping table, stop words are removed from a question and the remainder is sent to Internet search engines to obtain highly coherent sentences. Answer candidates not contained in the retrieved sentences are eliminated.

In the answer ranking phase, QFocus is used to sort the answer candidates derived from the Answer Extraction module. An answer candidate is given a ranking score if it fits the answer focus or limitations of the question. The candidate with the highest score is the one that fits the most clues of the question, and is therefore regarded as the top answer to the question.

3.4 QUESTION ANSWERING FOR HINDI

The paper in [46] focuses on developing Hindi QA system which has different language constructs, query structure, common words, and so on as compared to English. The main aim of the paper was to help elementary and high school students in getting correct answer for their subject related questions whereby facilitating e-learning in Hindi language.

For query construction, the researchers use self constructed lexical database of synonyms since there was no Hindi WordNet available. A case-based rule has been developed to classify questions. After the user question is changed to a proper query (by applying stop-word deletion, domain knowledge entity inclusion, and so on), the query will be submitted to the retrieval engine. Finally answer selection is done by extensive analysis of passages and the correct answer will be presented to the user.

The automatic Entity Generator: This module tries to recognize the entities in a particular course (domain specific entity) to which the user wants to pose questions. This configures the system automatically to any type of course domain. The system administrator on the server providing distance learning (or the user who wants to search answer from documents present in his local system) gives the

directory of files as input. The module then searches through the Main heading and sub-headings of the text files, recognizing them through their font size (larger than usual text size) and thus finds out the domain-specific entities. Word filtering is done to remove any elementary words. If no elementary words are found in the string then the whole string is also taken as an entity. The output is stored in the Entity file for subsequent use. This file contains domain specific entities.

Question Classification: Question classification is done by matching the patterns of interrogative words and questions are then put into the respective categories. The categories include questions seek reasoning, questions that require numerical data, questions that require an event, questions that require person, and so on.

Query Formulation: this module transforms the questions posed into proper query form that will be submitted to the document retrieval engine. The system uses the entity file to recognize the domain specific entities in the question. Individual keywords from the question will be compared with words generated from the domain specific entity files, hence, keywords with a match to domain specific entity keyword will be given maximum weight of 2, stop-words will be given 0 weight and normal term that didn't match the domain specific entity keyword will be given normal weight of 1.

Furthermore, the query will also be expanded with the self constructed lexical database for synonyms to match maximum number of documents for a question. Here, domain specific entity terms will not be expanded.

Answer Extraction: an IR engine (SEFT – Search for text IR) has been used to extract passages from a collection of documents. The answers to a query are locations in the text where there is local similarity to the query, where similarity is computed as the sum of weighted overlaps between terms. It was assumed based on intuitive notion that the distance between terms is indicative of some semantics of the sentence.

CHAPTER FOUR

THE AMHARIC LANGUAGE

In this Chapter we will first present an overview of the Amharic Language in light of QA. We will discuss the different structuring of Amharic Questions and Answers and outline challenges of QA in Amharic. We will also discuss the Amharic language number and ordinal representations which are the core concepts used in extracting numerical and date related answer particles. Finally, there will be discussion of Amharic punctuation marks that will help in separating sentences, so that sentence/paragraph indexing will be much easier.

4.1 GENERAL OVERVIEW OF AMHARIC LANGUAGE

Amharic is a Semitic language spoken in many parts of Ethiopia. It is the official working language of the Federal Democratic Republic of Ethiopia and thus has official status nationwide. It is also the official or working language of several of the states/regions within the federal system, including Amhara and the multi-ethnic Southern Nations, Nationalities and Peoples region. Outside Ethiopia, Amharic is the language of millions of emigrants (notably in Egypt, US, Israel, and Sweden), and is spoken in Eritrea [33]. It is written using a writing system called fidel or abugida, adapted from the one used for the now-extinct Ge'ez language.

Ethiopic characters (fidels) have more than 380 Unicode representations (U+1200-U+137F) [36]. The Unicode representation of Ethiopic characters is attached in Appendix E.

4.1.1 GRAMMATICAL ARRANGEMENT

We will not make a detailed explanation of the Amharic language's grammatical arrangements as it is beyond the scope of this study. We will cover the top level grammatical and morphological structure of the language in this section. The Amharic language has been declared to have word categories as **ስም** (noun), **ግስ** (verb), **ቅፅል** (adjective), **ተውሳከ ግስ** (Adverb), **መስተዋድድ** (preposition), and **ተውላጠ ስም** (pronoun) [22].

Noun: a word will be categorized as a noun, if it can be pluralized by adding the suffix **ኦች/ዎች** and used as nominating something like person, animal, and so on [24].

Verb: any word which can be placed at the end of a sentence and which can accept suffixes as **/ህ/,/ሁ/,/ሽ/**, etc. which is used to indicate masculine, feminine, and plurality is classified as a verb.

Adjective: any word that qualifies a noun or an adverb, which actually comes before a noun (e.g. **ጎበዝ ተማሪ**) and after an adverb (**በጣም ጎበዝ**). Other specific property of adjectives is, when pluralized, it will repeat the previous letter of the last letter for the word (e.g. **ረዥም--› ረጎዥም**).

Adverb: it will be used to qualify a verb by adding extra idea on the sentence. The Amharic adverbs are limited in number and include **ትናንት**, **ገና**, **ዛሬ**, **ቶሎ**, **ምንኛ**, **ክፉኛ**, **እንደገና**, **ጅልኛ**, and **ግምኛ**.

Preposition: preposition is a word which can be placed before a noun and perform adverbial operations related to place, time, cause and so on; which can't accept any suffix or prefix; and which is never used to create a new word. It includes **ከ፣ ለ፣ ወደ፣ ስለ፣ እንደ**…

Pronoun: this category further can be divided as deictic specifier, which includes **ይህ**, **ያ**, **እሱ**, **እስዋ**, **እኔ**, **አንተ**, **አንች**…; quantitative specifier, which includes **አንድ**, **አንዳንድ**, **ብዙ**, **ጥቂት**, **በጣም**…; and possession specifier such as **የእኔ**, **የአንተ**, **የእሱ**…

In question answering, part of speech tagging will have immense advantages to extract the exact answers. Factoid questions need answers that are most of the time nouns. Therefore, understanding the structure of a sentence whereby a given noun can be easily indicated will facilitate extracting the correct answer. The Amharic basic sentence is constructed from noun phrase and verb phrase (**ስማዊ ሀረግ + ግሳዊ ሀረግ**).

Sentence= noun_phrase + Verb_Phrase.

For example, the sentence: "***ሁለት ትልልቅ ልጆች ትናንት በመኪና ወደ ጎጃም ሄዱ።***" has noun phrase ""***ሁለት ትልልቅ ልጆች***" and verb phrase "***ትናንት በመኪና ወደ ጎጃም ሄዱ***".

Here we will not delve into the details of sentence structure and sentence parsing as we will use simple techniques of answer extraction approaches that is detailed in Chapter 6.

4.1.2 SENTENCES IN AMHARIC

In the previous section we have seen different categories and formation of words in Amharic. Here we will see details of Amharic sentences and their types based on the work in [24].

A sentence, in every language, is a group(s) of word(s) that comply with the grammatical arrangement of the language and capable of conveying meaningful message to the audience. A sentence in Amharic can be a **statement** which is used to declare, explain, or discuss an issue; an **interrogative** sentence

which can be used for questioning; **exclamatory** and **imperative.** We will discuss Amharic statements and interrogative sentence construction and structures which are the concern of this thesis work.

The statement (**አረፍተ-ነገር**) will have the noun phrase and verb phrase combinations. The noun phrase and the verb phrase further will be divided to different particles such as other sub noun phrase and verb phrase, noun, adjectives, specifier and so on. Similarly the interrogative sentence will have the same structure with little rearrangements and introduction of question particles. Questions will be raised for different purposes such as to know something unknown, or to assure something that is known.

4.1.3 QUESTION PARTICLES (INTERROGATIVE PARTICLES)

In every language, questions are constructed with the help of question particles (also known as interrogative words) and a question mark (?) which is placed at the end of the question. The question mark, by itself kept at the end of the statement, indicates that the sentence is a question. In English, the interrogative words (WH words) **who, what, where, when, why, how …** are used to construct a question [37]. In Amharic, there are a number of interrogative words that will help in constructing a question. Using sample questions collected from Radio and Television puzzles, most of it from [53] that happen to be prepared for Ethiopian Television Question and Answer Game as well as from Linguistic resources [34], we have identified question particles shown in table 4.1.

Table 4.1: Amharic Question Words

ማን, ለማን, ማነው, እነማን, ማንማን, ማንን, ማናማን, እነማንን, የማን, ከማንኛው, ማንኛው, ማንኛይቱ, ማንኛዋ, ማንኛቸው, በማን,
ተናገር, ጥቀስ, ግለፅ, አብራራ
የት, የቱ, በየት, የታ, የቲቱ, የቶቹ, ወዴት, ከየትኛው, የየት, የትኛው, የትኛዋ, የትኞቹ, ከየት, እስከየት, ወደ የት, የየትኛው, በየትኞቹ
ምን, ምንድን, የምን, ምንህን, ምንሽን, ምናችሁን, ምኔ, ምኑን, ስለምን, እንደምን, ለምን, በምን, እስከምን, ከምን, ወደምን, ምንጊዜ, ምንያህል,
መቸ, መቼ, በመቸ, እስከመቸ, ለመቸ
ስንት, ስንቱ, ከስንት,
ወይስ, ምረጥ, ወይ, ይሆን, እንደ, እንዴት

4.1.4 QUESTION AND ANSWER FORMATION

In English, most of the time, the interrogative words occur at the beginning of a sentence. For example, in the sentence "**Who** is the president of the USA?", the interrogative word **who** clearly indicates that it is a question sentence. The statement form of the question will be "**X** is the president of USA". In Amharic, however, the interrogative words are put at the end of the sentence more often than as the beginning. If we consider the question "**ከኢትዮጵያና ከኡጋንዳ ቀጥሎ በሶማሊያ የሰላም አስከባሪ ጦር በማሰማራት ሶስተኛዋ አገር ማን ናት**?", the interrogative word, **ማን** is placed near the end of the sentence. The statement form of the question could be "**ከኢትዮጵያና ከኡጋንዳ ቀጥሎ በሶማሊያ የሰላም አስከባሪ ጦር በማሰማራት ሶስተኛዋ አገር X ናት**". Same question in Amharic can be constructed in different ways. For example, if we are given a statement "**ታዋቂው ሴኔጋላዊ ድምፃዊና በአለም አድናቆትን ያተረፈው የሒፕ ሆፕ አቀንቃኝ ኤኮን ለኢትዮጵያዊቷ አማቹ ከ1 ሚሊዮን ብር በላይ የሚያወጣ የዳይመንድ ሠዓት ስጦታ ማበርከቱን አፍሪካ ፕሬስ ኤጀንሲ ሠሞኑን ዘገበ።**" we can ask same question in different form as

a. "**ታዋቂው ሴኔጋላዊ ድምፃዊና በአለም አድናቆትን ያተረፈው የሒፕ ሆፕ አቀንቃኝ ኤኮን ለኢትዮጵያዊቷ አማቹ <u>ምን ያህል</u> ብር የሚያወጣ የዳይመንድ ሠዓት ስጠ?**"

b. "**ታዋቂው ሴኔጋላዊ ድምፃዊና በአለም አድናቆትን ያተረፈው የሒፕ ሆፕ አቀንቃኝ ኤኮን ለኢትዮጵያዊቷ አማቹ <u>ስንት</u> ብር የሚያወጣ የዳይመንድ ሠዓት ስጠ?**"

Answers will be formed using some of the question terms and the expected answer or just the expected answer only. If a question is the type of yes/no, the answer can also be just yes/no, with possible explanations. If a question is the type of place name, the answer will be just the name of the place or a statement with the place name. For example, the question "**የኢትዮጵያ ዋና ከተማ ማን ይባላል**?" might have an answer "**የኢትዮጵያ ዋና ከተማ <u>አዲስ አበባ</u> ይባላል**" or just a short answer "**<u>አዲስ አበባ</u>**".

Linguists put the formation of Amharic questions and answers differently. Accordingly, questions can be raised about some action or condition, about the performer of an action, about the agent to perform the action or time and place, about the cause of the action or aim of the action, how the action is performed or techniques used to perform the action, and so on. In fact all of these types of questions occur in the phrase, so that the question focus is a phrase. Let us take an example "**ካሳ ከእናቱ ጋር በተሲያት ምሳውን በላ።**". Here we have two noun phrases (**ካሳ** and **ምሳውን**) and two prepositional

phrases (**ከእናቱ ጋር** and **በተሲያት**). We have also the verb phrase that is constructed from the noun phrases and the prepositional phrases which is **ከእናቱ ጋር በተሲያት ምሳውን በላ**. So the question will arise on these five phrases. On the noun phrase we can ask questions like "**ማን ከእናቱ ጋር በተሲያት ምሳውን በላ**?" and "**ካሳ ከእናቱ ጋር በተሲያት ምን በላ**?". So the question particles (question pronoun) **ማን** and **ምን** are placed on the place of the subject and object respectively. Similarly, questions can be asked as ""**ካሳ ከማን ጋር በተሲያት ምሳውን በላ**?" and ""**ካሳ ከእናቱ ጋር መቸ ምሳውን በላ**?". Here the questions are about the prepositional phrases. Finally, questions can be raised about the verb phrase as "**ካሳ ምን አደረገ**?".

In this study we will not apply deep linguistic analysis to analyze questions and extract answers formations. The techniques that we applied in analyzing and extracting correct answers are discussed in Chapter 6.

4.2 AMHARIC PUNCTUATION MARKS AND NUMERALS

The Amharic documents collected should be pre-processed before the succeeding AQA components perform further operations. The punctuation marks are here discussed as it will help in pre-processing documents. Sentence, paragraph, and document indexing all utilize different punctuation marks for separating one from the other. For example, sentence indexing will be done with the help of the Amharic full stop (**።**) for separating different sentences.

Similarly, numerals have greater impact on AQA systems. Since numbers are stored in different formats, there should be some kind of standardization that will help higher document relevance during searching. If Ethiopic and Arabic numbers can't be normalized to same standard, a document that has different number representation will remain irrelevant for document retrieval.

In Amharic, there are different punctuation marks used for different purposes [34, 35]. In the old scripture, a colon (two dots ፡) has been used to separate two words. These days the two dots are replaced with whitespace. An end of a statement is marked with four dots (**አራት ነጥብ** ።) while **ነጠላ ሰረዝ** (፣ or ፥) is used to separate lists or ideas just like the comma in English.

In Amharic, numbers can be represented using Arabic symbols. It has also its own number representations, Ethiopic number representations. Similarly numbers can be represented in a word alphanumerically. Table 4.2 shows the Arabic, Amharic and alphanumeric representation of numbers.

Table 4.2: Number Representations in Amharic

Arabic	Ethiopic	Alphanumeric	Arabic	Ethiopic	Alphanumeric
1	፩	አንድ	20	፳	ሃያ
2	፪	ሁለት	30	፴	ሰላሳ
3	፫	ሦስት	40	፵	አርባ
4	፬	አራት	50	፶	አምሳ/ሀምሳ
5	፭	አምስት	60	፷	ስልሳ/ስድሳ
6	፮	ስድስት	70	፸	ሰባ
7	፯	ሰባት	80	፹	ሰማኒያ
8	፰	ስምንት	90	፺	ዘጠና
9	፱	ዘጠኝ	100	፻	መቶ
10	፲	አስር	1000	፲፻	ሺ/ሽህ

In Amharic fractions and ordinals have their own way of representation [34]. Table 4.3 shows fraction and ordinal representations in Amharic.

Table 4.3: Amharic fraction and Ordinal representation

Fraction	Amharic representation	Ordinals	representation
½	ግማሽ	1st	አንደኛ/ቀዳማዊ
1/3	ሲሶ	2nd	ሁለተኛ/ዳግማዊ
¼	ሩብ/እርቦ	3rd	ሦስተኛ/ሳልስ
2/3	ሁለት ሲሶ / ሁለት ሦስተኛ	4th	አራተኛ/ራብዕ
¾	ሶስት-አራተኛ	.	.
1/10	አስራት	.	.
2X	እጥፍ	9th	ዘጠንኛ/ዘጠነኛ
2.X	ሁለት ነጥብ X	10th	አስረኛ

Dates in Amharic can be written in different ways. It can be written just with Arabic numbers like English dates such as 12/01/2001 or using Ethiopic numeral representation and alphanumeric representations.

4.3 Challenges in Amharic Questions and Answers

Questions in English will be constructed with question particles and most of them are not ambiguous. A question constructed with a question particle **where** definitely requires an answer of place types but never a person or a number. For example, the question “where is Lucy found?” requires a place name as an answer. Person name seeking questions will be constructed with the interrogative word **who** or

whom. When it comes to Amharic, it has very intrinsic problems related to the language. Consider the following questions [53]:

በቅርቡ በኢትዮጵያ የስራ ጉብኝት ያደረገ የፓርላማ ልዑክ ከየት ነው የተላከው?

ብሔራዊ የቡና ሙዚየም የሚገነባባት ከተማ ማን ናት?

የኢትዮጵያ ዋና ከተማ ማን ይባላል?

The question particle in the first question is **የት** which seems to expect place name as answer but actually the answer is name of an organization (**ከአውሮፓ ህብረት**). In the later two questions, the question particle **ማን** is used to request place name as an answer while it is customarily used to ask person name. From this point of view, we can say that the question particles in Amharic are multipurpose. The challenge here is that question particles, as of the English language question particles, will not help in pinpointing the expected answer types. This leads us to further investigate expected answer type identification techniques which will be explained in the subsequent Chapters.

Another very interesting challenge of QA in Amharic language is identifying names from verb derivation. Nouns in Amharic can be primitive or in connection with verbs and nouns (derived from verbs or from other nouns) [34]. Most Amharic names, especially person names are derived from verbs or just verbs which can be used as names. Using gazetteers to indicate named entity will face a problem of differentiating names from verbs. Names such as **አበበ**, **ከበደ**, **ለማ**, **አየለ**, **ተሰማ** are also verbs. In addition to verbs and person names, place names and person names such as **መሉ ጎጃም**, **ኢትዮጵያ**, **የጎጃም ወርቅ**… can also be interchanged that will make differentiating person name from place name difficult. We will see details of the problems faced and the techniques used to alleviate the problems in the implementation part of AQA (Chapter 6).

In English identifying proper name is not a problem as proper names are capitalized. There is no capitalization in Amharic so that a proper name will be written similarly with other parts of speech. Automatic named entity recognition in Amharic remains a very difficult task in the field of information extraction.

Statements in Amharic have unique punctuation mark, which is **አራት ነጥብ** (**።**), to separate from other statements. The problem occurs on the writing style of different bodies, where using two colons (::) in place of **።** will be very difficult to demarcate statements as these sybols have different Unicode representation. The other problem is that, there is a practice not to use the punctuation marks. Hence, there should be document normalization to bring the document to same standard.

4.4 SUMMARY

Amharic has unique characteristics compared with English and other languages. As information retrieval in general and question answering in particular has many language dependent techniques, studying the general characteristics of Amharic is mandatory.

Amharic has its own character (Fidel) representation. The grammatical arrangement of Amharic is also very different. Amharic has basic word categories as **ስም** (noun), **ግስ** (verb), **ቅፅል** (adjective), **ተውሳከ ግስ** (Adverb), **መስተዋድድ** (preposition), and **ተውላጠ ስም** (pronoun).

The question particles in Amharic are placed near the end of the sentence most of the time. Most of the question particles are multipurpose where a single question particle is used for different types of question formations. There should be extra information to determine the question type besides the question particles such as question focuses or complex grammatical structure analysis of the sentence.

Among the different types of punctuation marks in Amharic, Amharic full stop (።) is used to separate different statements. The question mark, just like the case of many other languages, will be placed at the end of interrogative sentences (questions). Numbers in Amharic will be represented in Arabic, Ethiopic, and alphabetical forms.

There are different challenges in Amharic question answering. One of the main problems is that question particles by themselves can't help in determining the question type. Extra analysis is required to determine the question type, so as to know the expected answer types. Secondly, some proper names belong to more than one word categories, such as verb and noun so that determining whether that word is the expected proper name or not is very difficult. This problem is aggravated as there is no proper name capitalization in Amharic. Lastly, statement demarcation is a problem as there is no standardized writing by different writers.

CHAPTER FIVE

DESIGN OF AQA (ተጠየቅ)

In this chapter we will discuss the architectural design of AQA, the main components and their interactions. We will first discuss the main components of AQA alongside the subcomponents and resources needed for each module/component. Finally we will show the entire architecture of AQA.

5.1 COMPONENTS OF AQA

Every question answering system will have basic components of Question Analysis, Document retrieval and Answer Extraction [26, 47]. The question Analysis component will be responsible in analyzing the question. It will perform tasks such as constructing proper query for the document retrieval component, determining the expected answer type and question types with possible question focuses, etc. The document retrieval component is responsible in retrieving the top N relevant documents that will be presented to the Answer Extraction component. The final component, Answer Extraction, is responsible in extracting the accurate answer to the user's question.

Though these are the basic components that every question answering system comprises of, the internal structures and algorithms of every QA system differs from system to system and from language to language. The technique and detail components of one question answering system to the other are quite different. It will be different based on the algorithmic technique applied, the NLP tools employed which is specific to that language, etc.

Hence, we will briefly describe the main components of AQA explored in this thesis work in details. In this study, we have identified five fundamental components: the document pre-processing, question analysis, document retrieval, sentence/paragraph ranking, and answer selection. Figure 5.1 shows the general architecture of AQA.

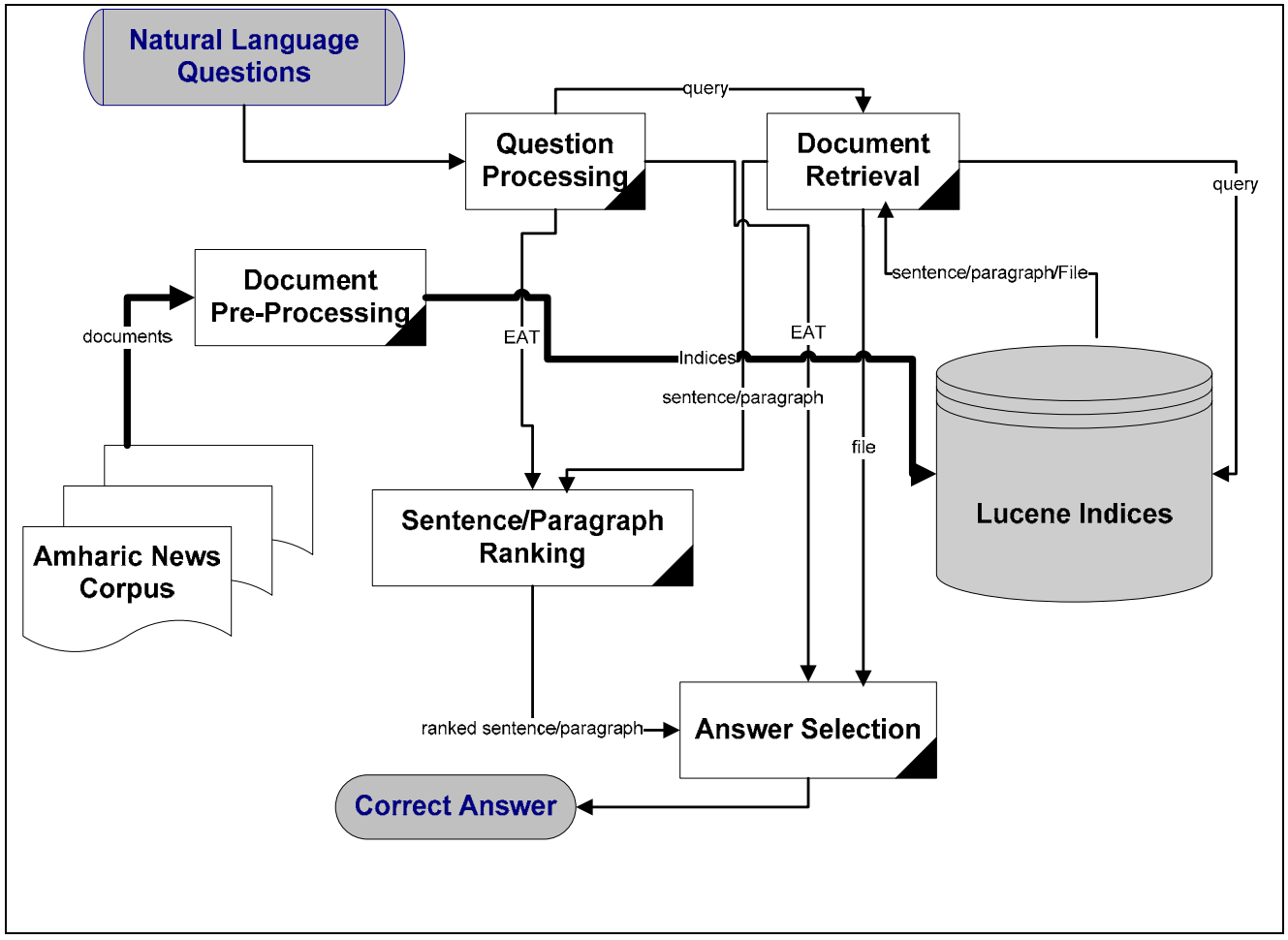

Figure 5.1: Architecture of AQA

The **document processing** module is required for normalizing language specific issues while the **question processing** component extracts the semantics and notions of the question so as to prepare meaningful query to the document retrieval component. The **document retrieval** component, based on the query generated by the question processing component, will retrieve documents that are relevant to the user question. While the document retrieval component is responsible in retrieving relevant documents, the **sentence/paragraph ranking** component will reconsider each document (such as sentence or passage) based on the requirement of the question to re-rank for the **answer selection** module to easily select the correct answer. We will address the details of each component in the subsequent sections.

5.2 DOCUMENT PRE-PROCESSING

The documents for this thesis work are Amharic news corpuses collected from different electronic newspapers over the last twelve years. Besides, for testing purpose, documents are prepared by hand that could be used to evaluate our system. The Amharic documents need different types of pre-processing before they are made ready for the AQA system. The main pre-processing techniques we have used are sentence/paragraph demarcation, Ethiopic number normalization, character normalization, sentence/passage based tokenization, stemming, stop-word removal, synonym, and gazetteer preparations. We will discuss them in details below. Diagrammatically the document processing components are shown in figure 5.2

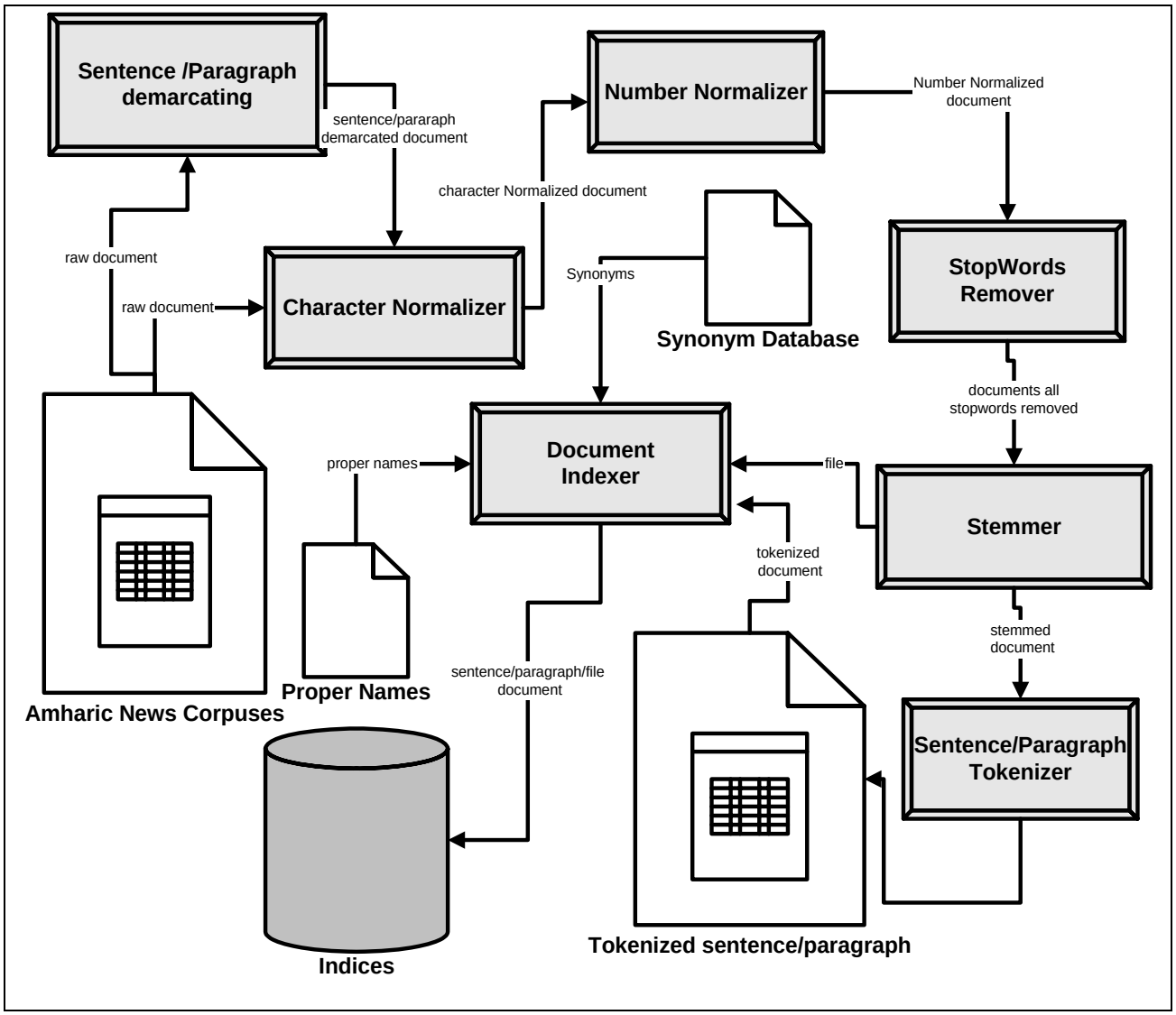

Figure 5.2: The AQA Document Pre-Processing Sub-System

Sentence/paragraph demarcation: Amharic statements are normally separated by special punctuation mark ። (**አራት ነጥብ**), the Amharic full stop. The electronic news corpus collected uses different formats of Amharic full stops. Some newspapers, such as Ethiopian Reporter, use two dots (፡ **ሁለት ነጥብ** - the Amharic word space), while others use two colons (:) instead of Amharic full stop. The most difficult part in sentence demarcation is when the corpus didn't use any of the punctuation marks. Here a special study has been made to demark a sentence in a given word boundary by considering special words or characters in a sentence. As it will be explained in Chapter 6, most sentences are demarked with the words **ነው**, **ታውቋል**, and characters X**ዋል**, X**ል**, etc where X is any character. Since answer extraction depends on a sentence, we have to incorporate a module to the AQA system that will normalize the document to one format, which is to sentences separated by Amharic full stop punctuation mark.

Paragraphs in Amharic normally start on a new line where the first line is indented with some spaces. But, the practice most news agencies followed is similar to the English paragraphs where it is not indented. In this thesis, paragraphs will be marked with some special symbol $ (a dollar sign to demarcate a paragraph).

Character Normalization: The Amharic Language has special characteristics where different letters with the same pronunciation and the same meaning can appear in a document. Tessema [15] has developed an analyzer for normalizing documents to a specific form of a letter such as **ሰ** and **ሠ** to **ሰ** and **ሀ**, **ኀ**, and **ሐ** to **ሀ** and **ጸ** and **ፀ** to **ፀ** as well as their orders (**ሠ**, **ሡ**, **ሢ**, etc. to **ሰ**, **ሱ**, **ሲ**, etc.) accordingly. In addition to this normalization, we further investigated and found that some other orders of the letters should also be normalized. For example **ሀ**, **ኀ**, **ሐ**, **ሃ**, **ኃ**, and **ሓ** should be normalized to **ሀ**. Similarly the characters **ቸ**, **ቼ**; **ሸ**, **ሼ**; **የ**, **ዬ**; **አ**, **ኣ**, **ዐ**, **ዓ**; **ው**, **ዉ**; **ኝ**, **ኜ**; **ች**, **ቺ**; **ጭ**, **ጪ**, and so on should be normalized to one form as they are being used interchangeably in documents.

Number Normalization: Numbers in Amharic documents can appear in different ways. As we have discussed in Chapter 4, Amharic numbers in a document can appear as Arabic numerals, letter numerals, or Ethiopic numerals (although rare in modern electronic documents). Numbers are normalized to different formats (especially Arabic and letter variations) and a query will be expanded that incorporates different representation of that number so that different representation of numbers in a document will be matched.

Stopwords Removal: Stopwords are removed from the document before the document is indexed so that trivial terms will not be considered for document retrieval in the latter stage of document retrieval.

Stemming: Stemming reduces words to their root word so that different variations of the root word will be matched to the root word during document retrieval. The stemmer developed in [15] is modified since it applies the algorithm on each and every word of the document. Normally proper names, dates, and numbers should not be subjected to stemming since they will not be reduced to root words. In this thesis work, list of proper names (Person name, Place names, Numerals, and dates) are prepared into a gazetteer. Our algorithm will not apply stemming to such words.

Sentence/passage based tokenization: once sentences and paragraphs are demarcated with the Amharic full stop and $ respectively, the next step is to chop up the document and the paragraph which will be ready for indexing using the Lucene API. The Lucene API will store each sentence and paragraph of a document as a separate Lucene document.

Synonym Indexing: Synonym indexing is very helpful especially in the absence of WordNet to a language [46]. We have identified synonyms that should be incorporated to the Lucene index that will help matching the larger variation of a query.

Paragraph/Sentence/Document Indexing: The last stage of document pre-processing is document indexing. Once the document contents are normalized using the previous techniques, it is indexed using the Lucene API. Sentence indexing is done on the document based on the sentence punctuation mark, Amharic full stop. Paragraphs will be indexed separately using the special paragraph markers ($$). Finally the whole document (a single text file) will be indexed in a different Lucene index. The three indexes will be used for document retrieval and evaluated based on the correct answer returned by the answer extractor component in a later stage.

5.3 QUESTION PROCESSING

Once documents are pre-processed and stored in a file system as a Lucene index, the next step is to process the user question for that will help in generating a well structured query. The question analysis module is the component that actually interprets the question the way it should be convenient for the document retrieval and answer extraction components. The question posed by the user will be processed before submitted to the IR component. Queries in a search engine and in QA are quite different in that the query in question answering needs extra information to be included (expanded) or removed. This component of AQA is considered substantially important; otherwise the remaining

components will be ineffective in retrieving correct documents and extracting an exact answer. The question analysis module produces a logical query representation, an indication of the question focus (answer clue words), and expected answer types [7]. Therefore, the question processing component of AQA has subcomponents such as question classification, expected answer type determination, and query generation. The details of these sub-components are discussed below. Figure 5.3 shows the subcomponents and their interaction.

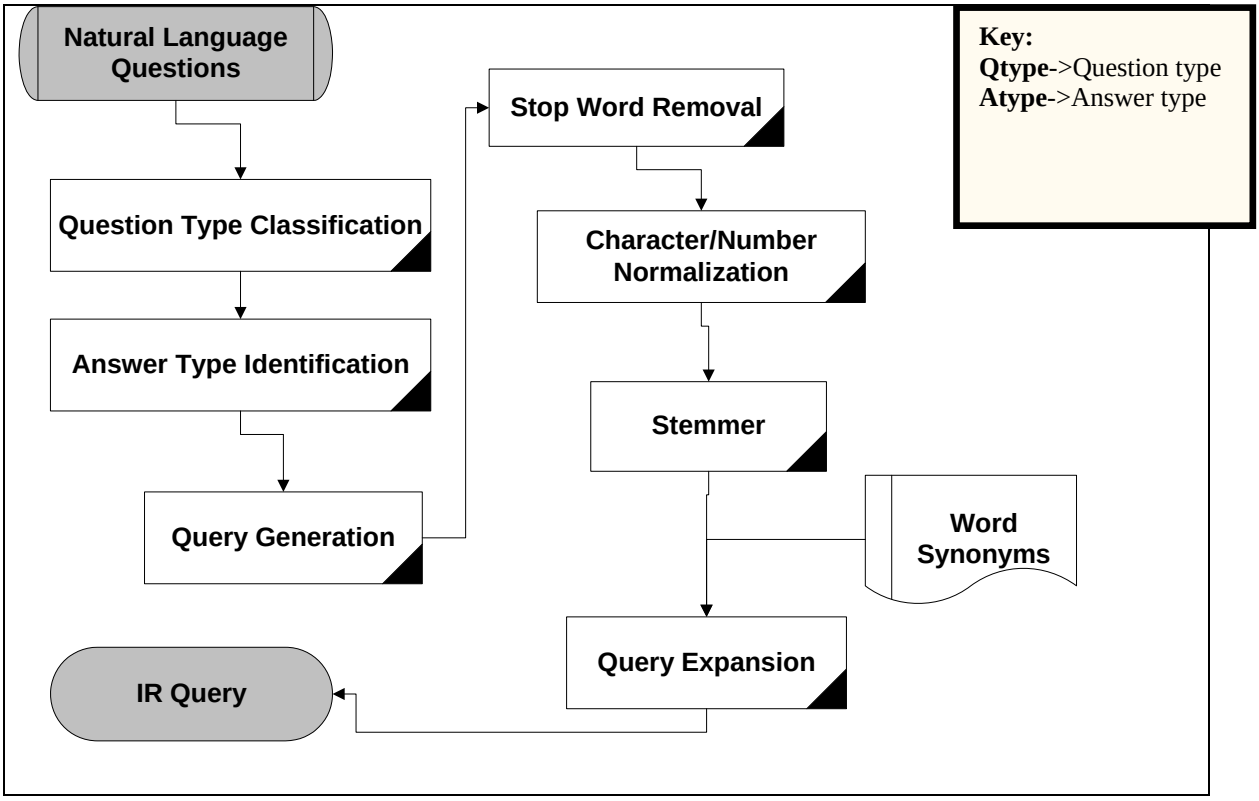

Figure 5.3: Question Analysis Subcomponent of AQA

Question Type Classification: The user query will be first processed to find out the type of the question it belongs to. The question classification component is a very important component of question analysis where the possibility of exact answers relies. If a question is classified wrongly in this stage, the subsequent stages potentially fail in extracting the correct answer. The detailed strategies and algorithms used in classifying the question types will be discussed in Chapter 6.

Answer Type Identification: Determining the expected answer type that the question seeks is also equally important as determining the question type. Actually, the answer types are directly related to the question types. The answer type to the question will be determined in this subcomponent that will be delivered to the sentence/paragraph ranking and answer extraction components of AQA. The

answer type determination component considers hierarchical answer types that are explained in Chapter 6.

Query Generation: The query generation component is responsible in generating a well formatted query for the IR component of AQA. The query generation component is very specific compared with the queries of search engines [7]. The query generation component will incorporate sub components such as stopwords removal, stemming, character normalization, etc. which have been used during document pre-processing so that the query and the document belongs to the same standard. The stopwords removal component will remove stopwords including the question particles in the question. The stemming component, which is actually the same component employed in document processing component, will reduce the query terms into their root words. The stopwords removal, stemmer, and character normalization components should be similar with the one used during document pre-processing module otherwise relevant document retrieval will not be up to our expectation. The query expansion subcomponent will include some extra terms (synonyms) which are used to boost document retrieval related to the question based on the question type and the expected answer types.

5.4 DOCUMENT RETRIEVAL

The document retrieval component of AQA is similar to many QA document retrieval components. The document retrieval component uses the Lucene API with some modifications. In addition to the core Lucene API components, we have used some of the Lucene contrib[†] packages such as RegexQuery for regular expression based searching. The document retrieval component, accepting the query from the question processing component, uses a combination of the Vector Space Model (VSM) of Information Retrieval and the Boolean model to determine how relevant a given Document is to the query [43]. The specific issues considered in this component are RegexQuery, SpanNearQuery, and Document Boosting.

RegexQuery: We have incorporated RegexQuery to apply the regular expressions developed specifically for date and numeral data retrieval. We have first identified patterns for date and numeric answer particles and the pattern will be passed to the document retrieval component with the help of RegexQuery. For example, if the question is of type birth date, then the RegexQuery will accept all the query terms plus the pattern of birth dates to retrieve birth date related documents. Therefore,

[†] The Lucene sandbox contains contributions to the core API such as language specific analyzers, WordNets, etc.

documents will be matched against the regular expression to retrieve relevant documents besides the query terms. It will positively affect the relevance of a document with higher probability of answer particle present in the documents.

SpanNearQuery: The SpanNearQuery matches spans (terms) which are near one another. The slope (the distance between terms) will help to determine how far the two terms should be in order to be considered relevant. The SpanNearQuery technique, together with the RegexQuery, helps specifically how far the query terms and the expected answers should be to be considered relevant. In addition to this, the SpanNearQuery specifies which terms are very important for document matching and documents which have all those terms will be considered relevant.

Document Boosting: Some terms will be highly indicative of the required answer than others. Those terms will be given higher boost value so that sentences containing those terms will be given higher scores. Question focuses and named entities are chosen to accept greater document boosting value.

5.5 SENTENCE/ PARAGRAPH RANKING

The sentence/paragraph ranking component of AQA ranks sentences or paragraphs according to the question types and expected answer types. In the case of sentences, the already ranked sentence returned by the Lucene IR system will be re-ranked based on the expected answer types [49]. The Lucene IR component returns documents with higher word overlaps or coverage based on the query terms. It is highly probable that the sentence returned by Lucene as atop ranked results in “No Answer”. This is because the Lucene API does not consider the answer types in its internal similarity scoring except in the case of RegexQuery where rules to match expected answer type can be passed together with the query terms. The sentence/paragraph ranking module has subcomponents like checking possible answer particles in a sentence, calculating the likelihood of a sentence, and returning the top n reranked sentences or paragraphs.

Checking possible answer particles: This subcomponent of the sentence/paragraph ranking module looks for answer particles in a sentence based on the question types and expected answer types. Hence, a sentence for a question of type ***place*** and possible answer type of ***city*** will be analyzed for possible occurrence of cities in the sentence. A sentence with a good number of cities that have a match with proper question focuses and question terms will be given due consideration to be ranked atop so that a higher weight value to that sentence/paragraph will be assigned. Therefore, this subcomponent

penalizes sentences returned by Lucene atop which have higher word coverage but no expected answer type in them.

Re-Ranking Sentence/paragraph: this module will rank a sentence/paragraph according to its weight value given in the previous subcomponent. Once a sentence has been given a numerical value as a weight, the one which has higher value is considered most relevant and ranked atop. The ranked top n sentences/paragraphs will be delivered to the answer selection component.

5.6 ANSWER SELECTION

The final component of AQA system is the answer selection module. The goal of answer selection is to choose from a pool of answer candidates the most likely answer for a question. This module is responsible in selecting the best answer from the sentences. The AQA system uses techniques such as checking the expected answer type, analyzing the question focus (pattern) and re-ranking the sentence based on the term frequency metric and **query term – answer candidate** distance measures. The components of the answer selection module are depicted in figure 5.4.

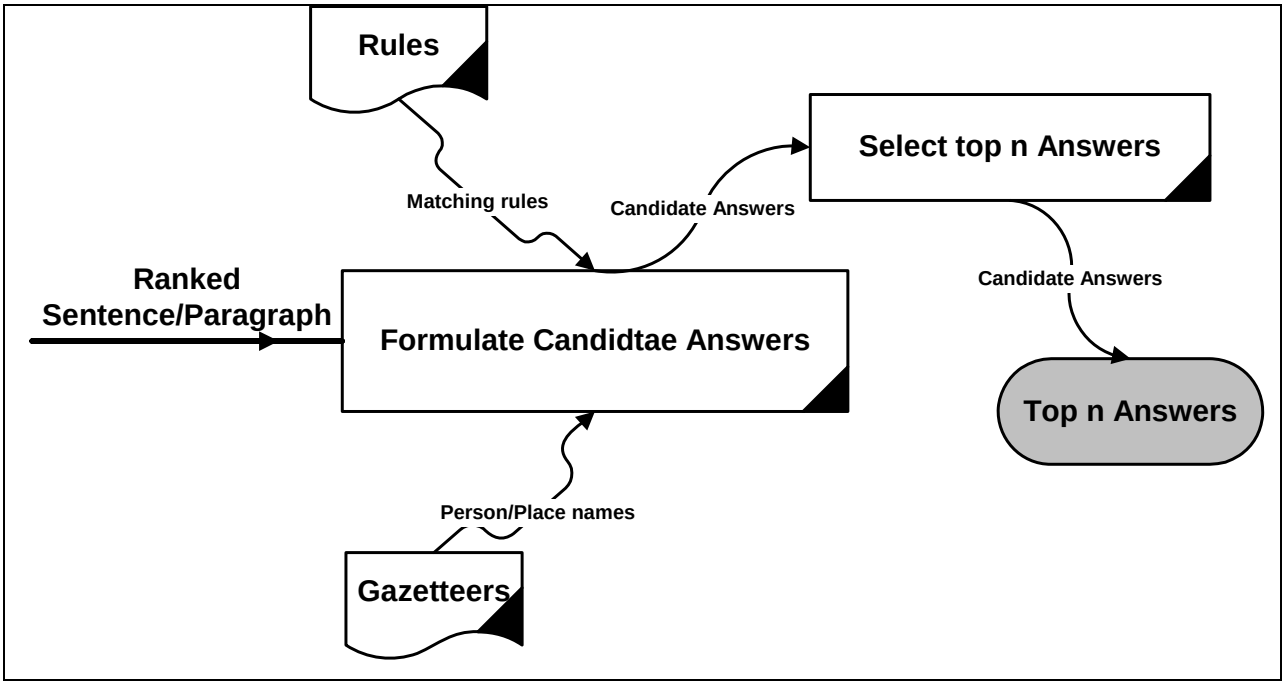

Figure 5.4: Components of Answer Selection Module

Extracting the correct answer based on the expected answer types and the question focuses have been used by many researchers [51]. The answer type of the question that has been the output of the question analysis module together with the possible question focuses is used in selecting the candidate answer. The techniques used in ranking the sentence/paragraph are also applied in the answer selection

stage. The rules and the gazetteers are incorporated to extract answers. Date and number type questions are matched with the rules developed and place name and person name type questions are matched with the gazetteers [50]. Hence, this module has subcomponents like formulating candidate answers and selecting the top n correct answers where n is 5.

5.7 SUMMARY

In this Chapter the architectural framework of AQA and the main subcomponents are discussed. The architecture of our system comprises of five main components. The document pre-processing component processes the document and makes it ready for retrieval. The question posed by the user will be accepted by the question processing module so that the question type, question focus, and expected answer type will be determined. The question processing module also generates the IR query to be submitted to the document retrieval component. The document retrieval component retrieves relevant documents and presents the result to the sentence/paragraph ranker module. Finally, the answer selection module selects the best answer to the user.

CHAPTER SIX

IMPLEMENTATION OF AQA: THE DETAILED DESIGN

In this Chapter, the detailed implementation of AQA will be presented. First we will cover the document pre-processing techniques and algorithms used in AQA. Next, we will explain the detailed strategies and algorithms implemented in analysing questions posed by the user so that the expected answer type, question type, question focus and proper query will be determined. The third section centers on the specific methods used in retrieving relevant documents from Amharic newspapers corpus of size about 15600 news articles. The fourth and fifth sections, which are the core part of this research work, detail the techniques, algorithms and strategies incorporated in re-ranking sentences or paragraphs and selecting the best answer among the pool of candidate answers respectively.

6.1 DOCUMENT PRE-PROCESSING

Amharic documents need further pre-processing to make them ready for the AQA system. The documents which are collected from different newspapers need extra efforts for normalization before the subsequent AQA subsystems act on them. We have incorporated this module for a number of reasons. One of the main reasons we have to process the documents is that the Amharic document behaves differently in pronunciation (reading) and writing that needs basic normalization. Unless documents are normalized to one standard of writing and reading style, document retrieval will severely be affected. The second main reason is, unless document normalization should be done, the performance of the system will be highly penalized. Therefore we have further broken down this section into three, that is, normalization to have same document format where documents exhibit differences in reading and writing, to boost document retrieval performance by making some kind of normalization, and indexing the document.

Section 6.1.1 discusses document normalization issues related to writing and reading. Section 6.1.2 addresses document normalization such as stemming, stop word removal, and so on. Section 6.1.3 discusses how the documents will be indexed and made available for retrieval.

6.1.1 DOCUMENT NORMALIZATION TOWARDS WRITING AND READING

In this Section we will present the techniques we have used to normalize documents that will create differences in writing style as well as reading accents.

Character Normalization: As discussed in Chapter 5, different characters (fidels) of Amharic are used interchangeably in writing and reading. Some of the characters such as **ሀ**, **ኀ**, **ሐ**, **ሰ**, **ሠ**, **ጸ**, and **ፀ** have been identified and normalized to one respective character class in [15]. In this thesis we have identified additional characters that we have found being used interchangeably in documents. The characters and their normalization is shown in table 6.1

Table 6.1: Character classes and their normalizations

Character classes	Normalized to
ቸ, ቼ	ቸ
ኾ, ኼ	ኾ
ሸ, ሼ	ሸ
የ, ዬ	የ
አ, ኣ, ዐ, ዓ	አ
ው, ዉ	ው
ኝ, ኚ	ኝ
ሽ, ሺ	ሽ

Hence, our algorithm normalizes the document by replacing every character classes, such as **ው**, **ዉ** to their normalized form like **ው**.

6.1.2 DOCUMENT NORMALIZATION FOR BETTER PERFORMANCE

Document normalization for character standardization helps to make a standard document that can match queries regardless of differences that occur during writing and reading. This will help in matching documents otherwise left unmatched. The other very important document normalization technique is to enhance performance of document retrieval and document matching. In this section we will see different techniques that will help to increase performance of document retrieval.

Stopwords removal: Stopwords are words that are frequently used so that they don't help distinguishing one document from the other, such as a, the, an, is, etc. for English [43]. We have modified the stopwords removal system developed in [15] to be used for the AQA system. All question particles such as **ማን** (who), **ስንት** (how many), **የት** (where), and so on will not be considered for indexing as they can't be used to match a specific document for retrieval. Therefore, all question particles that are identified are stopwords. Words like **ናቸው**, **ትናንት**, **ሰአት**, **ናት**, **ነበሩ**, **ነበረች**, **ወይዘሮ**, **ወይዘሪት**, and **አቶ** are considered as stopwords in the work in [15] but they are proved to be

goodwords* in our work as they facilitate document matching with expected answer type. In addition to these words, proper names, dates, and numbers are also considered as goodwords as they result in better document matching. Except the goodwords we have identified, all stopwords identified in [15] and all question particles are considered as stopwords.

Stemming: Stemming is the process of removing the commoner morphological and inflexional endings from words. Some researchers argue that for QA systems, morphological variants of words should be included with the query (query expansion) instead of stemming for better relevant document retrieval [23]. For our work we have applied stemming and didn't consider morphological variation searching except for number normalization expansion. Unlike character normalization, which is done by the document pre-processing module to make documents bear similar standardization, number normalization is performed at the question processing stage so that numbers show different number representation in the query will be expanded. The detail of number normalization is explained in Section 6.2.2. The stemming algorithm developed in [15] has been modified for our work. Instead of applying the stemmer for every word in a document, we have applied stemming only for non named entity terms. Proper nouns, dates, and numerals will be indexed as is, except with some prefixes and suffixes such as **በ**, **ለ**, **የ**, **ከ**, **ና**, **ን**, **ም**, **እነ**, etc.

Sentence/paragraph punctuation mark normalization: As discussed in Chapter 5, the collected Amharic documents are prepared using different punctuation marks for sentence demarcation. One of the normalization technique applied for sentence demarcation is to replace all groups of Amharic word space punctuation mark (**፡ ሁለት፡ ነጥብ**) and group of colon (:) occurrences with Amharic full stop (**። አራት፡ ነጥብ**). The other technique we have used is to demarcate sentences with some special Amharic sentence finishing word used to indicate end of a sentence. Words like **ነው**, **ነበሩ**, **ናቸው**, **ሆኗል**, **ታውቋል**, **ተዘግቧል**, are used as sentence finishing words. Therefore, documents that didn't use any punctuation mark are demarcated with the help of this sentence finishing words.

For paragraph demarcation, we fragmented documents based on paragraph separation with new line. If the documents are not separated with a normal paragraph separation, that is a new line followed by a blank line, we have counted 5 sentences to be grouped to make up a paragraph. Paragraphs and sentence demarcation will help to compare document retrieval for their best answer presence. The algorithm used to demark sentences and paragraphs is shown in figure 6.1

* Goodwords are words which have higher contribution in matching documents with the correct expected answer type [23].

For each document in a corpus directory //sentence demarcation

Check presence of punctuation marks ::(two colons) or ። (Amharic full stop) or ። (two Amharic word space)

If contains ::(Amharic full stop)

Do nothing

Else if :: or ። are detected

Replace with ። //The Amharic full stop

Else if nothing occur for the first 20 words //assume the max no of word in a sentence is 20

For each occurrence of ***ነው, ነበሩ, ናቸው, ሆንኗል*****……**

Add ***።*** *after each word // end of sentence demarcation*

End for

For each document in the corpus directory //paragraph demarcation

Check presence of ***blank line***

If present

Add paragraph marker ***$***

Else for each ***count*** *of five sentences*

Add paragraph marker ***$***

End for

Figure 6.1: Sentence/paragraph demarcations Algorithm

6.1.3 DOCUMENT INDEXING

The last process in document pre-processing is to index the normalized and demarcated documents as a Lucene index. The Lucene index will be stored with internal Lucene file format for later document retrieval components to access it. During indexing some special procedures such as boosting some documents based on the occurrence of some special terms (such as question focus and named entities), applying stopwords removal, and stemming will take place. Proper names, dates and numbers will make the document to have higher priority to be considered relevant so that boosting value will be given to the document to make the document match the query with higher score. Stemming will be

done on all parts of the terms except proper names, dates and numbers. All stopwords will be removed while goodwords will be given a higher boosting factor for the document containing these terms.

Likewise, synonym of terms will be incorporated during indexing to achieve larger likelihood of query matching during retrieval. Three separate indices will be created. The first will be indexing each file as a separate Lucene index. The second and third indices are sentence and paragraph indices based on the already identified punctuation marks used to demark sentences and paragraphs respectively.

6.2 QUESTION PROCESSING

The query for search engines and QA systems are different in such a way that queries in QA should be more specific so that locating relevant documents will be more predictive. The question will be first identified to which question type it belongs. Questions that are not factoid type will not be processed as it is beyond the scope of this thesis work so that "Non Factoid" will be returned. The factoid questions will be further classified to different question types based on the question particles in the question and the question focuses identified. This classification will help in locating exactly the correct answer by the later stage of the AQA system. Once question types, question focuses and expected answer types are determined, the next stage of question processing is to generate the proper query that will help in retrieving relevant documents. The query generated, which is based on the question types and the expected answer types, will be passed to the document retrieval component. Below we will explain the two main subcomponents of question processing which are Question Analysis (i.e. question classification, expected answer type and question focus determination) and Query Generation. First we will explain the question classification, expected answer type identification and question focuses determination techniques of the question processing subcomponent. Next we will present the detailed procedures employed to generate an AQA query which incorporates stopwords removal, character/number normalization, stemming and synonym word expansions.

6.2.1 QUESTION ANALYSIS

Questions will be first analyzed to determine the category the question belongs to, what will be the expected answer type, and the question focus it has. The question analysis module will determine the question types, the question focuses (if present) and the expected answer types. Question type identification involves techniques of identifying the question particles which will help in stating what the question is about. To do so, we have first identified the question particles in the question. Question

focuses are specific terms in the query that will tell about what entity the question is concerned. We have made detailed investigation to find out what specific terms are very important in telling what types of entities are sought after. The expected answer type, which is directly related to the question type and the question focus, tells which particular type of entity is sought after. We will discuss the details in question particle identification, classifying questions, and determining the expected answer types below.

Question Particle Identification: In Chapter 4, we have identified a number of Amharic interrogative words. Among those, we have identified some of them that are used to ask factoid questions such as place name, person name, quantity or numeric, and time questions. As we have already discussed, the question particles are used for different types of question formation. We will discuss the techniques how we have used those question terms to identify question types below. The identified question particles are shown in table 6.2

Table 6.2 Question Particles to determine question types

Question particles	Question types
ማን, **ለማን**, **ማነው**, **የማን**, **በማን** => (who, and whom variations)	Person name, place
የት, **በየት**, **ወዴት**, **የየት**, **ከየት**, **እስከየት** => (where and variations)	place
መቸ, **በመቸ**, **እስከመቸ**, **የመቸ**, **ለመቸ** => (when and variations)	Time
ስንት, **ስንቱ**, **ከስንት**, **በስንት**, **ለስንት**, **ምን ያህል** => (how many and variations)	numeric, time

Question particles by themselves will not fully assist in determining the question types as some of them are very general that can be used for more than one question type. For example, the word **ስንት** can be used both for time and numeric question types such as "**በኢትዮጵያ የእግር ኳስ ክለቦች ጨዋታ በስንት ሰአት ይጀምራል**?" and "**በታዋቂው አርቲስት ጥላሁን ገሰሰ ቀብር ላይ ስንት ሰው ተገኘ**?" respectively. Hence, question classification will not solely depend on question particles. Further investigation is needed such as determining the question focus.

Question classification: For this study, we have collected nearly 749 Question and Answer samples from [53]. In addition we have prepared about 300 questions that are formulated from Amharic newspapers. Although the questions from [53] are diverse types which include the extent of picture

and sound identification (e.g. **ይህች በምስሉ ላይ የምታይዋት ድምጻዊት ማን ትባላለች**?), the larger portions are factoid types. According to our finding, we have identified coarse categories of answer types for factoid question such as **ስም** (Proper Name), **ስፖ`ርት** (Sports), **ሙዚቃ** (Music), **ሳይንስ** (Science), **መጠን** (quantity), **ዓመት-በዓል** (holiday) and **ህገ-መንግስት** (constitution) as well as a larger groups of fine grained answer types. Appendix A shows the coarse and grained expected answer types that will help in classifying questions.

As an example, let us see classes of answer types for places that will help in classifying place related questions as shown in figure 6.2

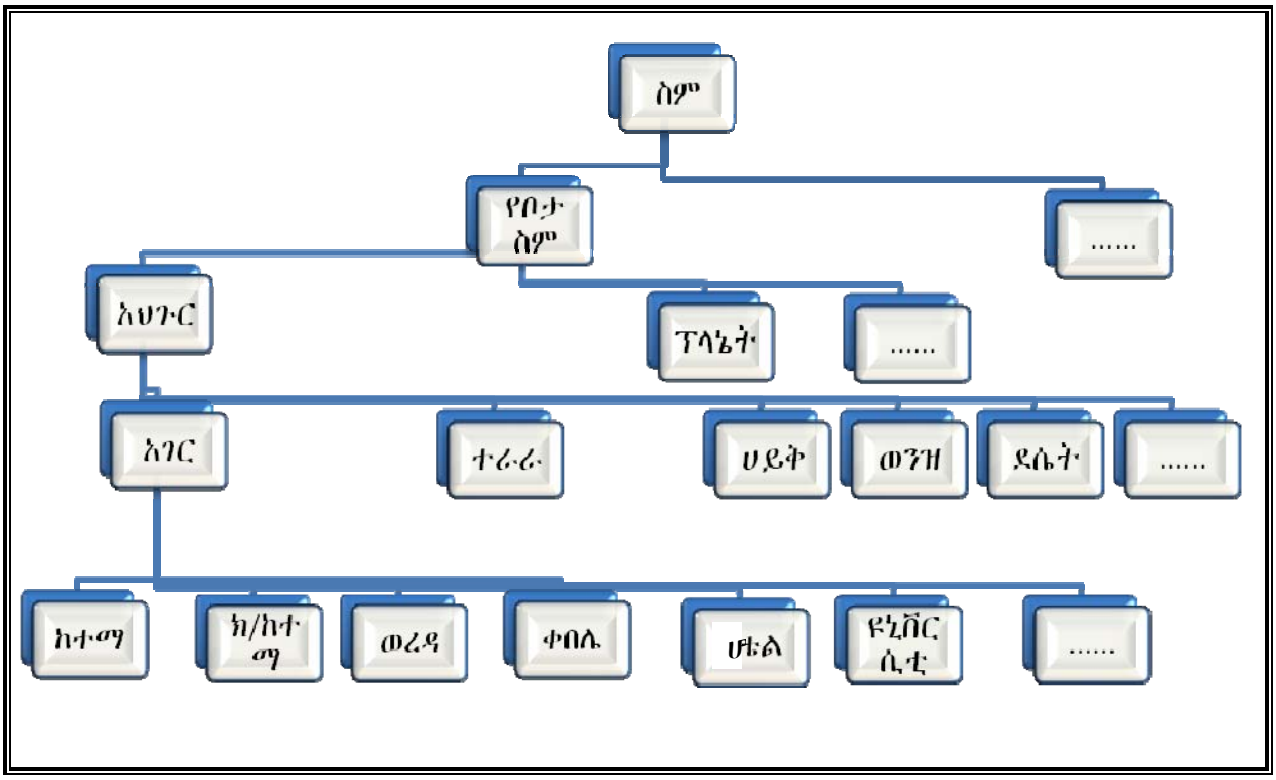

Figure 6.2: Answer types to classify questions

As it is shown in figure 6.2, together with question particles, we can easily determine the expected answer type for questions. Therefore, city question types can be easily recognized by the presence of question particles **የት/ማን** and question focuses such as **ዋና ከተማ**, **ከተማ**, **የክልል ከተማ**, etc. For example, the question "**የአማራ ክልል ዋና ከተማ ማን ይባላል**?" can be categorized as question type of city, with the help of question particle **ማን** and question focus **ክልል** and **ዋና ከተማ**.

Question Focuses: The question focus, as we have stated, are words or group of words that will give more hint about the question entity. Once again, with the help of sample questions and answers, we

have identified from [53] a number of question focuses that are learnt to be used automatically determine the question types and the expected answer types. Table 6.3 shows list of question focuses we have identified from the corpus.

Table 6.3: Question Focuses

Question Focus	Question Type	Expected answer type
የጠረፍ ከተማ/ከተማ/ዋና ከተማ/ክፍለ ከተማ…	City	City/town/country
አገር/ክፍለ አገር/ወንዝ …	Place (Country)	Country/city/town/mountain…
ቡድን/አሰልጣኝ/ዳኛ …	Team/person	Team/player/coach/instructor/…
ጠቅላይ ሚኒስትር	Person	Person
ፓርቲ	Organization	Organization/party/…
ማተሚያ/ቤት/ገበያ/ድልድይ/ቤተ-ክርስቲያን/መስጅድ/ት-ቤት/ ሆቴል...	Place (Entity)	Entity/item/
ሰአት/አመት/ወር/ደቂቃ/አ.ም	Time	Time/day/hour/month/year/minute…
ኪ..ሜ/ሜ/ርቀት/ሜ.ዋት/ብር…	Quantity	Number

Expected Answer type: Once the question types and the question focuses are determined, the next stage is to determine the expected answer type. Expected answer types will help to correctly locate the answer particles for extracting the exact answer by the answer selection and ranking modules. The expected answer types are directly related to the question types where the question type (either coarse group or grained group) will be mapped to it. If a question is of type place (coarse group), then the expected answer type will be a place too. To exactly pinpoint the expected answer type, we incorporated the question particle, question focus and the question type together. So given the question "**በንፋስ ስልክ ላፍቶ ክፍለ ከተማ በ** 98 **ሚሊዮን ብር ወጪ የተገነባው ሆቴል ማን ይባላል**?", the question particle, **ማን** indicates that the question type is place (but we need further information as this question particle can also be used to indicate person name as expected answer type) and the term **ሆቴል** (a question focus) further narrows down the expected answer type to be hotel. A simplified algorithm for determining the expected answer type is shown in figure 6.3.

For each natural language question posed
    Check for question particles **የት, በየት, ወዴት, የየት, ከየት, እስከየት**
    If question contains one of these question particles
        Classify question as **place type**
    Else check for question particles **መቸ, ለመቸ, እስከመቸ,** …
        If question contains one of these question particles
            Classify question as **time type**
    Else check for question particles **ስንት, በስንት, ከስንት,** …
            If question contains terms like **ቀን, ሰአት, ወር, አመት,**…//question focus
                Classify question as **time type**
            Else If question contains terms like **ኪ.ሜ, ሜ, ርቀት, ሜ.ዋት**
                Classify question as **numeric type**
            Else Classify question as **numeric type** and **time type**
    Else check for question particles **ማን, ለማን, በማን, ከማን**
            If question contains terms like **ከተማ, አገር, ተራራ**, … // question focus
                Classify questions as **place type**
            Else classify question as **person type**
    Else Try the IR based technique
End for

Figure 6.3: Question Classification Algorithm to determine expected answer type

Figure 6.3 clearly shows that questions will be analyzed first for the presence of possible question particles. Questions with none of the question particles already identified will be classified as can't be answered and no answer will be returned. Otherwise, it will first check the question if it has the specific question particles that will have clear indication on the question type such as **የት** and **መቸ** which always indicate place and time question types unambiguously. If the question contains ambiguous question particles such as **ስንት**, the question type will be identified after looking into the possible question focuses present in the question. Lastly a question with ambiguous question particles and unidentified question focus will be given a try for both type of question types, such as person and

place or time and numeric while it might result in a surprising answer and exponentially high response time.

Expected answer types can also be determined by IR system similarity scoring technique where sample questions with the expected answer types are indexed. For this research we have indexed some 300 questions with their expected answer types. This technique, while efficient and easier to develop, needs a lot of questions with different formats for higher precision. The IR system technique is used when the rule based technique fails to exactly determine the expected answer type and produces “no answer”. If our rule can’t determine the expected answer type based on the rules developed, we will use the IR system as a final attempt to find its expected answer type. If it is impossible to determine the expected answer type by both techniques, the question will be considered as non factoid and no further processing will take place.

6.2.2 QUERY GENERATION

The natural language questions posed by the user will not be submitted to the document retrieval component directly as it is. There should be some kind of modification that will maximize the probability of matching relevant documents. This subcomponent is very crucial as wrong queries might result in returning a wrong document where incorrect answer would be extracted. The query generation subcomponent includes stemming, character/number normalization, stopwords removal, and synonym expansion. Below, we will discuss each procedure that will help in generating the query.

The first task in query generation is to remove all the stopwords including the question particles and punctuation marks. The algorithm used to remove stopwords is the same with the one we have used in the document pre-processing module. The stopwords removal component is integrated in this subcomponent as stopwords are already removed from the index and will not help matching relevant documents. Once the stopwords are removed, the remaining query terms will be subjected to character normalization that will help in changing the characters to same format used in the document pre-processing module. If character normalization is not considered in this stage of question processing, relevant documents remain unmatched with the query.

The other very important task in query generation is number normalization. Number normalization will be done in the question processing module to match various representations of numbers in a document. During document pre-processing, we did not make number normalization to preserve natural

coherences of documents. That means, if we make number normalization to documents, then all documents will show similar number representation where it might be illogical. This will help the user to enjoy the natural variation of number representation in Amharic. If we consider number representations of hundreds, thousands, millions, and billions, they are normally expressed alphabetically than numerically. For example 2 ነጥብ 3 ቢሊዮን ብር is formally used in Amharic documents instead of 2,300,000 ብር. Hence, the main task of number normalization in query generation is that the users' question which contains numeric particles will be normalized so that the different variations of that number will be included to the query (query expansion).

Stemming will be applied to the query to reduce terms to their root words. Once every query term is stemmed, the synonym of every word will be checked from the synonym database to include synonyms of words to the query using the Boolean OR operator by the query expansion component. Figure 6.4 shows a detailed scenario how a query is generated.

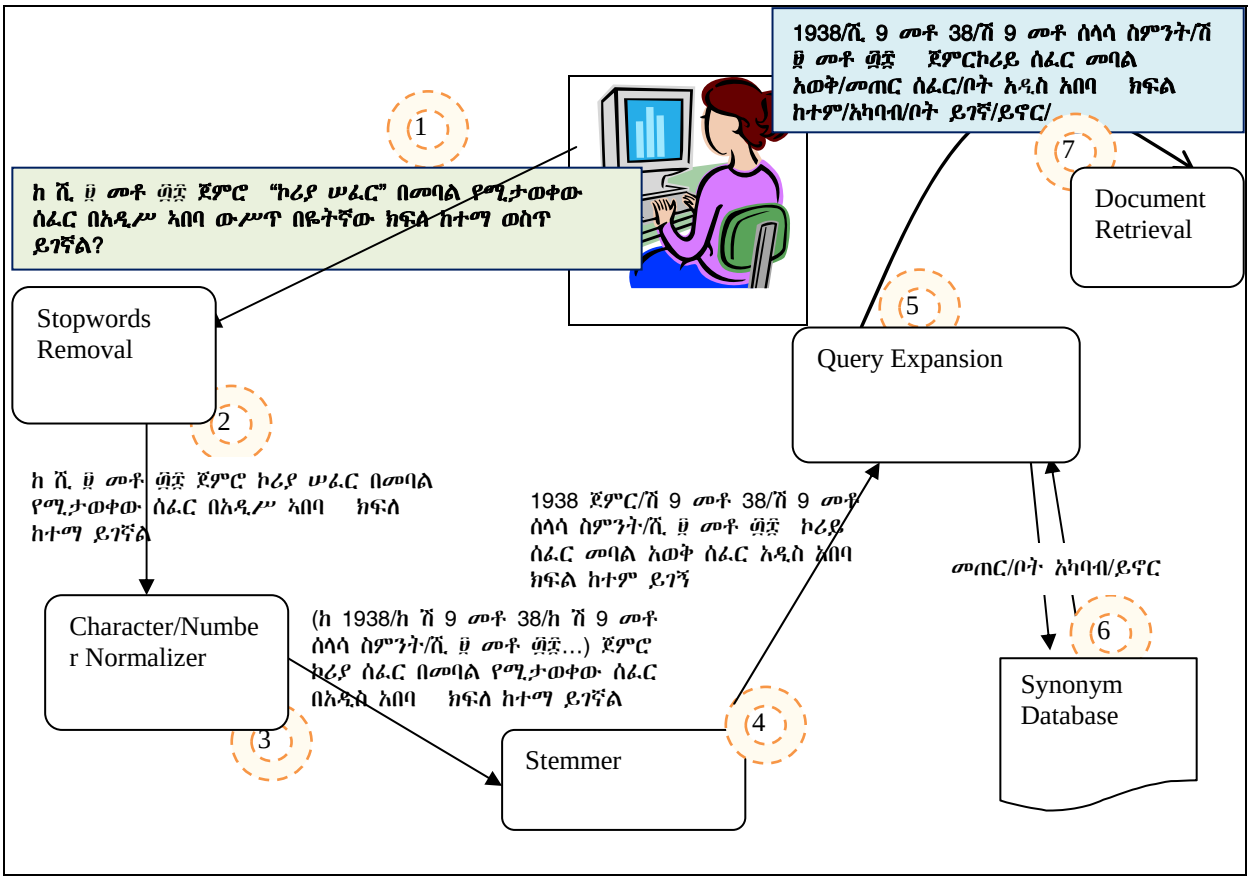

Figure 6.4: Scenario for Query Generation

Number 1 in figure 6.4 shows the question that is posed by the user. The stopwords removal subcomponent, as indicated in 2, removes the stopword **ውሥጥ**. The character/number normalization module at 3 normalizes the characters such as **ሠ** to **ሰ** and the number **ሺ፱ ፱፻ መቶ ፴፰** is normalized to include 1938 and **ሺ 9 መቶ 38**. The stemmer at 4 reduces words to their root words and submits the result to the query expansion module. The query expansion module checks synonym of all terms from the synonym database and includes the synonyms to expand the query (5, 6). Finally the generated query is submitted to the document retrieval module (7).

6.3 DOCUMENT RETRIEVAL

Generally, QA systems often contain an information retrieval subsystem that identifies documents where the answer to a question might appear [30]. In this thesis work, a **document** means that a file, paragraph, or a sentence that Lucene stores in its internal file system. It can be argued that the quality of the document retrieval module is highly related to the accuracy of the QA system as an irrelevant document never produces a correct answer [54]. There are different techniques considered appropriate for document retrieval of QA systems. The ad hoc strategy employed for document retrieval in QA systems is just to use the traditional IR techniques used in search engines. The AQA system uses three different types of document retrieval techniques that help answer selection modules to extract the correct answer. As documents are pre-processed and indexed in three formats, we can use sentence, paragraph, and file based document retrieval. In the case of sentence based retrieval, the Lucene API will just calculate the similarity of a query and a sentence based on its internal similarity scoring function. Sentence base document retrieval has shown a very significant effect on correct answer selection if there is a good match with the query terms and a sentence and an answer particle presents. The problem with sentence based document retrieval is that the exact answer and the query terms might be distributed in more than one sentence in which case the answer will be fragmented in more than one sentence and hence selecting the correct answer is problematic. As an example, consider these two sentences:

> **”የኢትዮጵያ ፕሬዝዳንት ማርማ ወ/ጊወርጊስ ስለአገሪቱ አጠቃላይ የልማት እድገት መግለጫ ሰጡ። ባለፈው ግንቦት** 17 **ቀን** 75**ኛ ልደታቸውን ያከብሩት ፕሬዝዳንቱ…. ”.**

If we ask a question “**ማርማ ወ/ጊወርጊስ ስንት አመታቸው ነው**?”, we can easily see that the answer can’t be extracted from a single sentence.

The above problem can be solved by paragraph retrieval where the paragraph normally encloses more than one sentence. The paragraphs that are demarcated by the special symbol $ sign in the document pre-processing module will be matched based on the calculation of the term similarity and the one with the better score will be returned. The problem we have encountered with the paragraph retrieval technique is that, while it has better answer space coverage, it might produce more number of candidate answers in a paragraph which needs extra semantic analysis of that paragraph to determine the correct answer.

The file retrieval technique, that is matching the whole file with the query, is a little bit different than the sentence/paragraph retrieval technique where the whole file content will be subjected to answer particle analysis by the later modules. Most researchers use this technique to retrieve the relevant document but there will be complex and overwhelming answer extraction techniques to determine the exact answer. Below, we will see these three (sentence, paragraph, and file) document retrieval techniques in detail.

DOCUMENT RETRIEVAL TECHNIQUES

The Lucene internal document scoring technique will be used for comparing sentences/paragraphs/files with the query. Hence, a document with more number of terms matched with the query terms receives higher score. The Lucene similarity function computes the score of query q within a document d as follows [43]:

$$Score\ (q, d) = \sum_{t\ in\ q} (tf(t\ in\ d).idf(t).boost(t.field\ in\ d).lengthNorm(t.field\ in\ d)$$

Where the factors in the scoring function described in table 6.4

Table 6.4: factors in the scoring function

factor	Description
tf(t in d)	Term frequency factor for the term (t) in the document (d).
idf(t)	Inverse document frequency of the term.
boost(t.field in d)	Field boost, as set during indexing.
lengthNorm(t.field in d)	Normalization value of a field, given the number of terms within the field. This value is computed during indexing and stored in the index.

When documents are retrieved, question focuses will be given higher boost value to make documents with the question focus more relevant. Hence, queries with specific question focus will be boosted by a value of 3 (a value we leant through experimentations); that is documents which contain this specific question focus will have thrice score value than the others. Therefore, if the question is about date types such as "**አርቲስት· ጥላሁን ገሰሰ በስንት· ዓ.ም. ተወለደ**", then documents with **ዓ.ም**., **ተወለደ**, and **አርቲስት·** will be given three times higher priority than the other documents.

The other technique we have used for document retrieval is incorporating special Lucene APIs such as RegexQuery and SpanNearQuery. The RegexQuery functionality is used to incorporate regular expressions within the query terms. The query will be matched, in addition to the normal query terms, with the designed regular expressions. Suppose the question is about specific date of birth. The query term will contain regular expressions such as ***"((19|20)\\d\\d), ((0[1-9]|1[0123])[-/.](0[1-9]|[12][0-9]|30)[-/.], etc."*** so that documents will be matched with these regular expressions besides the normal query terms. We have used RegexQuery specifically for numerical and date question types.

The SpanNearQuery functionality is more meaningful for this research to specifically determine how far the query terms should be spaced from each other to be considered relevant. SpanNearQuery is used to filter out documents which have very less number of query terms where they potentially result in No answer or false answer. The normal Lucene API will match every query term with documents and considers a document as relevant if the document contains at least one query term. Most importantly, the SpanNearQuery determines whether every query term is present in the document with the specified slop value. A slop value is the number of terms allowed between query terms in a document to be considered relevant. Hence, SpanNearQuery improved relevant document retrieval as documents irrelevant but contain the query terms in a longer distance from each other will be penalized. For example, if the question is "**የአባይ ወንዝ መነሻ የት· ነው**?" with the query "**አባይ/ናይል ወንዝ መነሻ/መጀመሪያ/ ነው/ይባላል**" and if the slop value is set to 5, documents with these query terms only in a distance of at least 5 words, both in forward and backward direction, will be considered relevant.

In addition to all the above document retrieval techniques, we have also considered the maximum number of query terms a document should contain to be considered relevant. Based on different experiments done for checking document relevance, we have formulated the following rules in determining the number of query terms a document should contain to be relevant.

- ***Rule one****: If the number of query terms generated by the query processing module is less than three, then a document should contain all the terms to be considered relevant.*
- ***Rule two****: If the number of query terms is greater than three and less than six, at least 3/4 of the query terms should be present in a document to be considered relevant.*
- ***Rule3****: If the number of query terms is greater than seven, then the document should contain at least 2/3 of the query terms to be considered relevant.*

These above designed rules show significant improvements for sentences and paragraphs, as the number of terms in a sentence or a paragraph is small so that maximum document relevance will be accomplished using these rules.

6.4 SENTENCE/PARAGRAPH RE-RANKING

The document retrieval component of AQA returns relevant documents that are believed to contain the correct answer to the question. The main research component we have actually developed is on the correct answer extraction technique. For IR systems, such as search engines, the task of the system is completed when relevant documents are returned. The document retrieval component, as discussed in the previous section, returns sentences/paragraphs/files based on the query term similarity scoring function. For search engines and other IR systems, it is up to the user to select the best answer by going through the documents one by one till the required information is obtained. In that case, the user may get the required information from the first document or has to go one by one through all documents till the required information is accessed. The case for QA systems is different in that the analysis task where correct information (i.e., the answer) sought after remains the main assignment of the QA system. The sentence/paragraph ranking module of our system will analyze every document returned by the document retrieval component for possible occurrence of answer particles. As we have already discussed in Chapter 5, a document ranked atop may have least possibility or none to have the expected answer type in it. Let us consider how a query and a sentence with 100% word coverage results in "No Answer". For the question "**የኢትዮጵያ ህዝብ ብዛት ስንት ነው**?", and a sentence " **የኢትዮጵያ ህዝብ ብዛት ከጊዜ ወደ ጊዜ እየጨመረ ነው።**", as we can see, the query have 100% word coverage with the sentence so that the Lucene API will return this sentence as a top result while it is not containing the expected answer. Hence, just as users discard this sentence as unimportant (actually after reading the whole document), the AQA system will do the same for extracting the best answer. In this Section we will discuss the procedure we have developed to re-rank documents that might help in

extracting the correct answer. The first task will be checking whether every sentence or paragraph bears an answer particle. Those sentences that didn't contain any answer particle will be removed from the list. Next a weight will be given to each sentence or paragraph based on the distance of the answer particle(s) and their query term in the sentence or paragraph. That means, if we take a query and a sentence/paragraph which has the expected answer(s) in it, the answer particle which have the smallest distance (number of non-query terms between the answer particle and the actual query term(s)) and more occurrences of the query term will be given higher rank. Below are details of sentence/paragraph re-ranking techniques we have considered for this research work.

6.4.1 ANSWER PARTICLE PINPOINTING

Answer particles in a document should be located before the re-ranking module considers the document for processing. A document which does not bear any answer particle is considered irrelevant and should be discarded. Based on the question type and the expected answer type determined in the question processing module, we have two types of answer pinpointing techniques. If the question is type of person and place questions, the document will be checked for the presence of possible place and person answer particles with the help of gazetteers. The problem with gazetteer based answer particle pinpointing is that some named entities are used as multipurpose in a part of speeches. Consider the entity **ታደሰ**, this term can be named as a person name (most commonly) and can also be named as place name as **ታደሰ ገበያ** and considered as a verb in some other occasions as **ሕንፃው ታደሰ**. The other problem is that the gazetteer is not capable of including all named entities. The second technique is pattern based or rule based answer pinpointing. Documents will be subjected to the rules we have developed to locate possible answer particles. The pattern based answer pinpointing technique includes dates, numeric, and person name question types. The rules for date and numeric question types are based on the regular expression matches. The rule for person name is based on title identification techniques. Accordingly, if possible answer particles are present in the document based on gazetteers and pattern pinpointing techniques, the document is considered relevant and presented to the re-ranking and answer selection module for further processing; otherwise, the document is discarded. The regular expression developed for date and numeric question types are shown in table 6.5.

Table 6.5: Rules for numeric and date question types

Question types	Rules
Numeric	"\\b(([በከየለውና])|አንድ|ሁለት|ሶስት|አራት|አምስት|ስድስት|ሰባት|ስምንት|ዘጠኝ|አስር|አስራ|ግማሽ|ሩብ|ሀያ|ሰላሳ|አርባ|አምሳ|ሀምሳ|ስልሳ|ሰባ|ሰማኒያ|ዘጠና|መቶ|ሺ|ሺህ|ሚ[ልሊ.][የዮ]ን|ቢ[ልሊ.][ዮየ]ን|ት[ሪሊ.የ]ን|አንደኛ|ሁለተኛ|ሶስተኛ|አራተኛ|አምስተኛ|ስድስተኛ|ሰባተኛ|ስምንተኛ|ዘጠነኛ|አስረኛ|ኛ|\\d|\\.|ነጥብ|\\s)+(\\s+)(በላይ|በታች|አካባቢ|ብር|ዶላር|ስኩየር|ኪሎ|ሜትር|ዩሮ|ኪ..ሜ")*\\b"
Date	Rule1="(\\b(ነሀሴ|መስከረም|ጥቅምት|ህዳር|ታህሳስ|[ጠጥ]ር|የካቲት|መጋቢት|ሚያዚያ|ግንቦት|ሰኔ|ሀምሌ)\\b\\s*([1-9]|0[1-9]|1[0-9]|2[0-9]|30)\\s*\\b(ቀን)\\b(\\s)*([,/.፣])*(\\s)*((19|20)\\d\\d))|(\\b(ነሀሴሜ|መስከረም|ጥቅምት|ህዳር|ታህሳስ|[ጠጥ]ር|የካቲት|መጋቢት|ሚያዚያ|ግንቦት|ሰኔ|ሀምሌ)\\b\\s*([1-9]|0[1-9]|1[0-9]|2[0-9]|30)(\\s)*([/,..፣])*(\\s)*((19|20)\\d\\d))|(ነሀሴ|መስከረም|ጥቅምት|ህዳር|ታህሳስ|[ጠጥ]ር|የካቲት|መጋቢት|ሚያዚያ|ግንቦት|ሰኔ|ሀምሌ)(\\s*)(\\d)*(\\s*)(ሰኞ|ማክሰኞ|ረብኡ|ሀሙስ|አርብ|ቅዳሜ|እሁድ|ቀን)*\\s*(([1-9][0-9])\\d\\d)*" Rule2="((0[1-9]|1[0123])[-/.፣](0[1-9]|[12][0-9]|30)[-/.፣](19|20)\\d\\d)" Rule3= "\\b((19)\\d\\d)\\b(\\s)*([,/.-፣])*(\\s)*\\b((20)\\d\\d)\\b" Rule4= ((ዛሬ|ዘንድሮ|ትናንት|ሰኞ|ማክሰኞ|ረብኡ|ሀሙስ|ቅዳሜ|እሁድ|አርብ|እለት)\\s*|([በለከ])((([0-9][0-9])\\d\\d)|\\d+)+\\s*| [ከበለ]* (አንድ|ሁለት|ሶስት|አራት|\\d*|አምስት|ስድስት|ሰባት|ስምንት|ዘጠኝ|ባለፈው|ሚቀጥለው|መጭው)+\\s*(አመት|አመታት|ሳምንት|ሳምንታት|ወር|ቀን|ቀናት|ሰአት|ደቂቃ|ሰከንድ|ወራት|ሰአታት|ሰኞ|ማክሰኞ|ረብኡ|ሀሙስ|ቅዳሜ|እሁድ|አርብ)\\s*(በፊት|በኋላ)*\\s*|([ከበለ]*አንድ|ሁለት|ሶስት|አራት|አምስት|ስድስት|ሰባት|ስምንት|ዘጠኝ|አስር|አስራ)+\\s*(አመት|አመታት|ሳምንት|ሳምንታት|ወር|ደቂቃ|ሰከንድ|ቀን|ቀናት|ሰአት|ወራት|ሰአታት)\\s*)+"

As it can be seen from table 6.5, we have one block of regular expression to match all different variations of numbers in a document. It can even match numbers with extra identifiers such as **ከ 65 ሺህ ብር በላይ**, **አንድ ነጥብ አምስት ቢሊየን ዶላር**, 876 **ስኩየር ኪሎ ሜትር**, and so on. For the date question types, there are four different types of patterns. In this case, documents are matched in the order the regular expressions are mentioned, that means if the document contains the string "**አርቲስት ጥላሁን ገሰሰ መስከረም** 17 **ቀን** 1934 **ተወልዶ**….", then the regular expression will match the longer match first, i.e., **መስከረም** 17 1934, not **መስከረም** or 17.

For person name types, we have also developed an alternative pattern based answer pinpointing technique. The pattern for person name is based on title matching. We have identified list of titles that precede a person name. This technique is very useful for foreign person name pinpointing. Besides, the gazetteer we have developed didn't include all possible person names and some of the names in the gazetteer are multipurpose that leads to wrong person name pinpointing. List of titles that we explored from different documents are indicated in table 6.6

Consider the following example:

If the question is "**የትግራይ ክልል ርዕሰ መስተዳድር ማን ይባላሉ**?" and the following document is found, then with the help of the title **አቶ** the answer **አቶ ፀጋዬ** will be detected.

Document:

……የትግራይ ክልል ርዕሰ መስተዳድር አቶ ፀጋዬ በርሄ፣ የአማራ አቶ አያሌው ጎበዜ፣ የኦሮሚያ አቶ አባዱላ ገመዳ፣ የደቡብ አቶ ሽፈራው ሽጉጤ፣ የቤኒሻምጉል ጉሙዝ አቶ ያረጋል አይሸሹም፣ የጋምቤላ አቶ ኡሞድ ኡቦንግ፣ የሐረሪ አቶ ሙራድ አብዱልሃዲ፣ የሶማሌ አቶ አብዱላሂ ሐሰን፣ የአፋር ክልል ምክትል ርዕሰ መስተዳደር መሐመድ ሻሂር፣ የአዲስ አበባ ከተማ አስተዳደር ከንቲባ ኩማ ደመቅሳ እና ምክትላቸው ከፍያለው አዘዘ አትሌቶቹ ባስመዘገቡት ድል የተሰማቸውን ደስታ ለመላው የኢትዮጵያ ህዝብ አስተላልፈዋል።

Table 6.6: 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, ባላምበራስ, 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, አፈ ጉባኤ, የተከበሩ, አምባሳደር, ኮማንደር, ብርጋድየር ጀኔራል, ሌተናል ኮሌኔል, ሼም, 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,

6.4.2 SENTENCE RE-RANKING

Sentences are the smallest document elements where determining the possible answer particles will be much easier. A sentence, for example, observed for the presence of person name, will be examined against the gazetteer/pattern to determine if it contains possible answer particles. If a sentence bears no answer particles, it will be automatically declared irrelevant and will be discarded. Otherwise, it will be considered relevant and again will be subjected for weighting. The problem comes if a given sentence contains more than one answer particle to calculate the weight of that sentence so that only one answer particle should be selected. In this case, the sentence will be computed against both

answers and the weight will be given based on the least distance calculation with the query terms. Let's consider this sentence:

በዚሁ ስብሰባ ወቅት የኢፌዴሪ ፕሬዚዳንት አቶ ግርማ ወልደ ጊዮርጊስ ምክር ቤቱ ባፀደቀው የድጋፍ ሞሽን ላይ የላኩትን የምስጋና መልዕክት የተከበሩ አፈ-ጉባኤ አምባሳደር ተሾመ ቶጋ ለምክር ቤቱ በንባብ አሠምተዋል።

The user may pose a question as "**የኢትዮጵያ ፕሬዚዳንት ማን ይባላሉ**?" that will be changed to a query like "**ኢትዮጵያ/ኢፌዴሪ ፕሬዚዳንት ይባላል/ይታወቃል/ነው**…" that have expected answer type of Person name (president). Here two expected answer types, **አቶ ግርማ ወልደ ጊዮርጊስ** and **የተከበሩ አፈ-ጉባኤ አምባሳደር ተሾመ ቶጋ** are detected as a person name. Hence, our algorithm will calculate the distance of each query term ("**ኢፌዴሪ** and **ፕሬዚዳንት**") from expected candidate answer **አቶ ግርማ ወልደ ጊዮርጊስ** and **የተከበሩ አፈ-ጉባኤ አምባሳደር ተሾመ ቶጋ** respectively. The distance value for each term and the total distance calculated are shown in table 6.7.

Table 6.7: sample distance calculation for the query "ኢፌዴሪ ፕሬዚዳንት"

Terms	Distance from	
	አቶ ግርማ ወልደ ጊዮርጊስ	**ተሾመ ቶጋ**
ኢፌዴሪ	1.0	17.0
ፕሬዚዳንት	0.0	16.0
Total distance	**1.0**	**33.0**

As it can be seen from table 6.7, it will be clear that **አቶ ግርማ ወልደ ጊዮርጊስ** is very near to the query term so that the weight of the sentence will be 1.0. According to our findings, most answer particles are very near to the query terms compared with other multiple answer particles. The algorithm to assign weight for each sentence based on the distance of the term from the candidate answer(s) is shown in figure 6.7

*For each **query terms** and **expected answer types (EAT)** accepted from the query generation subcomponent of question processing*

a. *For each sentence*

i. *Find **answer particle(s)** based on EAT and count the number of answer particles (**count**)*

ii. *If **count > 0** go to **iii**. Else go to **vii**.*

iii. *for each **answer particle** present*

1. *For each **query term***

a. *Count the number of terms between the **answer particle** and itself*

b. *Add the value of each query term distance (**distancecount**)*

iv. *If still more answer particles, go to **iii**.*

v. *If count=1//only one expected answer*

1. *Return the answer particle*

vi. *Else if count >1, return the answer particle with the least distance (**distancecount**)*

vii. *Else if count = 0, discard the sentence //no answer particle presents*

viii. *If the selected answer particle*

*Count the number query terms in a sentence **(countqueryterm)***

*Assign **countqueryterm** as a weight to the answer particle*

End for

For all sentences with the identified answer particle

*Sort the sentences based on their weight//based on **countqueryterm***

End for

Figure 6.5: Algorithm for determining sentence weight

The algorithm in figure 6.5 first identifies the answer particles from the sentence (if any) based on the distance between the answer particles and the query terms. Answer particles found very near to the

query terms will then be stated as candidate answers from that sentence. Once all candidate answers are selected from each sentence, the re-ranking will be done based on the number of query terms in the sentence. The answer particle which is found from a sentence with more number of query terms will be given the highest rank.

6.4.3 PARAGRAPH RE-RANKING

Paragraphs are considered for our thesis work due to the nature of concepts being distributed in more than one sentence so that the answer will not be discovered from one sentence. Sometimes, to get the correct answer, it will be mandatory to navigate sentences as a concept toward the answer extends over those sentences. Paragraphs are specially important if the exact answer is mentioned somewhere in a sentence while the succeeding sentence talks about the same concepts where Lucene tries to match every sentence even if the sentence does not contain the expected answer. Consider this example:

……ምክር ቤቱ ከዚህ በተጨማሪ በዛሬው ውሎው በንግድና ኢንዱስትሪ ጉዳዮች ቋሚ ኮሚቴ ለኢትዮጵያ ስኳር ኢንዱስትሪ ልማት እንዲውል ከህንድ መንግስት የተገኘውን የተጨማሪ ብድር ለማጽደቅ የወጣውን ረቂቅ አዋጅ በአብላጫ ድምጽ አፅድቋል። ብድሩ ያስፈለገበት ምክንያት በአገሪቱ የሚገኙ ትላልቅ የስኳር ኢንዱስትሪዎችን ለማስፋፋትና ሌሎች አዳዲስ ፋብሪካዎች ለመክፈት እንደሆነ በሪፖርቱ ተገልጿል። በዚህም መሰረት አጠቃላይ የብድር መጠኑ 640 ሚሊዮን የአሜሪካ ዶላር ስምምነት ሲሆን የሚከፈልበት የጊዜ ገደብ 20 ዓመትና የብድሩ ወለድ መጠንም 1 ነጥብ 7 በመቶ ብቻ እንደሆነ ተብራርቷል።

Now, if the user puts a question "**ኢትዮጵያ ለስኳር ኢንዱስትሪ ልማት ማስፋፊያ ከህንድ ምን ያህል ገንዘብ ተበደረች**", the expected correct answer is available only at the third sentence where primarily the first sentence matches the query but with "No answer". The algorithm used in ranking the sentence has been modified so that we have used it to re-rank the paragraph. The problem with paragraph re-ranking is, as we have mentioned previously in sentence re-ranking, there might be a number of expected answer types even which are near to the query term to give 'false' least distance value. The best work-around solution we have found is to determine the most important term in the question (i.e., question focus) and in the paragraph that should be considered for re-ranking. So, answer candidates that are very near to the question focus have been given higher weight compared to the other query terms. In addition to this, as we will see in the answer selection Section, answers which repeat themselves in more than one paragraph will be substantially important and will be given higher value.

Once the candidate answer is selected from each paragraph, the number of query terms in the paragraph and the number of candidate answers repeated in the paragraph will be used as re-ranking criteria. The candidate answer that is repeated more than once and with maximum number of query terms in the paragraph will be ranked on top.

6.4.4 FILE RE-RANKING

Files are text documents that have been returned by the document retrieval component. Files are different from sentences and paragraphs in that a given file may incorporate a number of answer particles as well as a number of query terms in the document. Re-ranking files have been found less effective as a file, most of the time, will contain the exact answer unlike sentences and paragraphs so that the rank Lucene computed is maintained. That means, files returned by the document retrieval component will be examined for answer selection in the order they are returned. Hence, we have developed a technique that will be applied to select the correct answer in the best efficient manner than incorporating the re-ranking algorithm used for paragraph and sentence re-ranking. The specific algorithm designed for answer selection in a sentence/paragraph/file is presented in detail in Section

6.5 ANSWER SELECTION

The final stage of AQA system is answer selection. The sentence/paragraph ranking module has ranked the sentence/paragraph according to the presence of answer particles and the distance of query terms from the answer particles. The answer selection module will extract the best answer from the pool of candidate answers found in the ranked documents. In this research, we have developed different techniques to select the best answer. The ranked sentence/paragraph by the sentence/paragraph re-rank module as well as the file will be again analyzed by the answer selection module to select the best answers. Once documents are given weight and ranked by the re-ranking module, the next task is to check whether an answer particle is repeated in more than one document or not so that the rank of the answer particle in the document will be recalculated, that is the sum of the two answer particles, hence the answer particle will have a higher weight. The idea is, since documents are collected from the Web, the exact answer for a given question might be repeated in more than one document so that wrong answers that have already received a higher weight because of the higher query term coverage will be prohibited from being the correct answer. If we consider a question that needs the name of prim a minister of a country, then, most probably the name of the prime minister

will be mentioned in more than one document so that the rank will be higher. When checking repetition of answer particles, we have also considered a function called ***contains,*** where short form of an answer such as "***መጋቢት*** 12 ***ቀን***, 2001 ***አ.ም***" and "***መጋቢት*** 2001" will be considered as same answer particles. This technique specially helps in selecting person name where the full name of the person will be mentioned once in a document and only the short form of the name will be repeated in the remaining content of the document such as "***ጠቅላይ ሚኒስትር አቶ መለስ ዜናዊ***" and "***አቶ መለስ***".

Counting the occurrence of answer particles in more than one document helps selecting the best answer where a possible answer particle is believed to occur in more than one document. The other technique is to consider the first answer particle from the ranked documents. The benefit of this technique is that sometimes correct answers about a specific issue might be sited in one document perfectly and none in other documents. In this case, even if Lucene returns a number of relevant documents by only considering fragments of the query terms, these documents might not contain the correct answer. In the following subsections, we will discuss the two answer selection techniques: by counting the multiple occurrences of answer particles and by selecting the top answer particles.

6.5.1 ANSWER SELECTION BY COUNTING MULTIPLE OCCURRENCE OF ANSWER PARTICLE IN SENTENCES/PARAGRAPHS

In this technique, the main assumption is that the correct answer will be repeated in more than one sentence that matches the query. The idea is an answer particle most of the time will occur with each query term so that a sentence that was ranked atop with possible wrong answer will not be selected as an answer because it will occur in more than one sentence. A sentence/paragraph sometimes may match with the query but with the wrong answer which has the least distance from the answer particles so that it might be selected as a candidate answer. Consider this example:

…….**እንዲሁም አቶ ዘርዓይ አስገዶም የቀድሞው የውጭ ጉዳይ ሚኒስቴር አማካሪ የነበሩት *የኢትዮጵያ ሬዲዮና ቴሌቪዥን ድርጅት* ዋና ሥራ አስኪያጅ** ሲሆኑ **አቶ ያረጋል አይሸሹም የፌዴራል የጎበረት** …… Here we can see that **አቶ ያረጋል አይሸሹም** will be considered as correct answer for the question "**የኢትዮጵያ ሬዲዮና ቴሌቪዥን ድርጅት ዋና ሥራ አስኪያጅ ማን ይባላሉ**?" as it has least distance from the query terms while it is wrong. Fortunately, the correct answer will repeat itself in more than one sentence/paragraph if there are large amount of corpus to search on for questions. The

algorithm for selecting the answer particles that occur in more than one sentence as the best answer is shown in figure 6.6.

*While there are extra sentences containing a **candidate answer***
*For a **candidate answer** in a sentence*
***Count** the number of query terms in the sentence //count*
For a candidate answer in a sentence
Compare the candidate answer with candidate answers in other sentences
*If it matches with other candidate answers (or **contains** function)*
Add count of the two answer particles
Concatenate the two sentences for summary
Remove the sentence from the list
End for
End while
For each weighted sentence //selection
For each sentence (concatenated sentences) compare its weight with succeeding sentence(s)
If the weight of the current sentence(s) is less than the succeeding sentence
Swap their position
Return the highest count sentence
End for
End for

Figure 6.6: Selecting a sentence based on the higher occurrence of a candidate answer

6.5.2 ANSWER SELECTION BASED ON SENTENCE/ PARAGRAPH RANK

The idea of counting answer particles repeated in more than one documents and consider the one that is repeated more often is a noble idea. The problem arises when the exact answer is sometimes present in only one sentence and the query term matches in more than one document. There is a probability, in the collections of the corpus, that the specific answer particle is mentioned once while actually the Lucene API returns hundreds of results based on the popularity of the query terms. This technique is very efficient if the fact being searched is available in the corpus and only in one document. If the

previous technique is used for such kind of questions, wrong answers repeated in more than one document will be considered as an exact answer. Hence, this will guarantee us that false answer particles repeated more than once couldn't be considered as best answer. The evaluation of the two techniques is discussed in Chapter 7.

6.5.3 ANSWER SELECTION FROM A FILE

As we have discussed in the previous Section, Lucene API will return a sentence, a paragraph or a file. Answer selection from a file is not trivial as the file comprises of a number of paragraphs and sentences. In the case of files, the answer particle(s) might be distributed throughout the document. If the question raised is a type of date, we can get tens of candidate answers of expected answer type date. Hence, it is highly probable that the answer selection module will return a wrong answer from these pool of expected answer types unless special consideration is made. The advantage of file based answer selection is that the answer will be for sure available somewhere in the document unlike the sentence based answer selection technique where concepts distributed over a range of sentences might miss the exact answer. For this research work we have used a technique to find the correct answer, that is selecting the best answer which is very near to the query terms based on the query term similarity calculations. The idea is similar with the technique we have used in determining the correct answer particle in the sentence and paragraph re-ranking module. The difference here is the query terms can be far apart from each other throughout the document and the answer particles also might be too far from each other. What we have done is that, the answer particle (which probably is also repeated throughout the document many times) which is very near to the query terms (once again the query term might also be repeated many times in the document) will have higher probability to be the correct answer. Most often, the answer particle also can be distributed over a range of documents where the probability of being the correct answer is higher. Therefore, two similar candidate answers extracted from two different documents will be the most probable correct answer than the one which is obtained from a single document, and the one which is repeated twice or more in a single document will be the most probable candidate answer from that document. Hence, the technique we have developed is that first the most probable answer will be extracted from a document and creates a candidate answers pool. The candidate answers pool will be again checked for duplicate candidate answers where the one which is repeated more than once together with its weight (the candidate answer placed atop has always the higher rank) will be the correct answer to the question. When the candidate answers are

selected from every document, we have assigned a weight for each of them which is calculated as follows. First the document's score obtained by the Lucene internal similarity function will be taken. Then the number of query terms in the document will be multiplied by the score to get the modified score of the document. Therefore, candidate answers from the document will be stored in the candidate answer pool along with its score. When checked for duplication, the candidate answers' score will be added and multiplied by 2/3 (taking the average has down weighted the rank so that we take 2/3 instead). Hence, among the pool of N candidate answers, the first 5 candidate answers along the document will be returned to the user. The algorithm that is used to extract a candidate answer and count the occurrence of each candidate answer is shown in figure 6.7

```
For each file returned by Lucene API //selecting the best answer from a document
        For each answer particle in the document
            For each query term in the document
                Count the number of words between the term and the answer particle
            End for
            Add the distance of all terms to the answer particle
            Return the answer particle with minimum distance
        End for
End for
For each candidate answer in a candidate answer pool
        Get the internal Lucene score of its parent document
        Count the number of query terms in the document //count
        Multiply the score with count
        Return the new score
End for
For each ranked candidate answer
    Compare the candidate answer with the rest candidate answers (apply contains rule)
      If duplicate found
        Add the two scores, multiply by 2/3
End for
For each candidate answer
     Select the first 5 candidate answers to the user
End for
```

Figure 6.7: Finding the best answer from files

6.6 SUMMARY

The AQA system implementation consists of five main modules. The **document pre-processing** component is used to normalize documents to a given standard. The Amharic document normalization includes character normalization, number normalization, punctuation normalization, stopwords removal, stemming, and synonym indexing. Once documents are normalized and indexed, they will be ready for the succeeding components to further process and extract correct answers.

The **question processing** component of AQA is used to manipulate the questions to create a proper query term, expected answer types, question types, and the question focus. The query generation subcomponent of question processing is used to create a proper query that will be submitted to the document retrieval component. Question types are determined so that determining the expected answer type will be easier. The question focus determination improves document retrieval and expected answer type identification. The expected answer identified will facilitate the document ranking and answer selection modules.

The **document retrieval** component is responsible to retrieve relevant documents to the latter AQA modules. Sentences will be retrieved from the sentence index of Lucene to extract the correct answer at sentence level. Further, to make the answer coverage wider, paragraphs are considered so that the possibility of finding the correct answer will be higher. To the wider possibility of answer coverage, we have incorporated document (text file) retrieval so that finding a correct answer in one document will be maximized. While sentence-based answer retrieval is efficient for documents where full factual information are to be found in one sentence, it is less effective for documents where concepts encompass a number of sentences. To the contrary, file level retrieval shows better probability of answer selection from top documents while performance will be severely penalized. Paragraph retrieval can be moderate in the case of performance and efficiency. Evaluation of all techniques is discussed in Chapter 7.

Sentence/paragraph re-rank module of AQA is used to re-rank sentences/paragraphs based on the availability of answer particles, number of query terms, and number of answer particles. Sentences/paragraphs with no expected answer particles in them will be automatically removed from candidacy. Sentences/paragraphs with more number of query terms will receive higher weight and ranked atop. Answer particles in a sentence/paragraph very near to the query terms will have higher opportunity to be a candidate answer.

Lastly, the **answer selection** module will select the best answer from the pool of candidate answers. Candidate answers which have higher weight (more number of query terms) and repeated in more than one sentence/paragraph will be considered as correct answers. Similarly answers in a file will be selected based on the least distance from the query terms. The more the number of query terms in a document, the more the answer particle repeats itself in the document and the more the answer particle repeated in more than one document, the higher the possibility of that candidate answer to be the correct answer.

While named entity recognition based answer selection techniques have been extensively researched in this thesis work, we have also considered rule-based answer selection techniques for questions that can't be answered directly by the named entity recognition technique. In the rule-based technique, rules i.e., patters for documents to match, have been developed to find some specific types of answer particles for some question types. The rules developed help in extracting foreign person names and place names where the person name and place name can't be found in the gazetteer list.

CHAPTER SEVEN

EXPERIMENT OF AQA (ተጠየቅ)

ተጠየቅ ስፋኖው ?
ለአሁን ማንነትህ ለክብርህ ለሞተው
ፋሽስትን ደምስሶ ድንበር ላስከበረው
……
ተጠየቅ ስፋኖው እኮበል ንገረው ?
እንዲያ ነፍሱን ከፍሎ ሀገር በነፃነት ክብር ላቆየው
……
ተጠየቅ ስፋኖው እኮ በል ንገረው ?
አንድ ክፍለ ዘመን 100 **ዓመት ምን ሰርተህ ቆየኸው?**
……
ሁሌም ይከበራል ዘላለም ይወራል
ለአንተስ የሚወራ ምን ታሪክ ሰርተሀል?
(**መልዕክተ ዮሐንስ**፤ 2001)

This chapter focuses on the evaluation of our system, its performance, reliability and trustworthiness. Every QA system will be evaluated towards its effectiveness, mainly correctness, completeness and exactness with recall and precision computations. The TREC workshop held in November 1999, included a “track” on question answering where the goal was to evaluate technology for finding answers to fact-based questions in unrestricted domains.

Our QA system has been given a name ***ተጠየቅ (Be questioned),*** a historical verbalism in Ethiopia where two people appear before a judge used to ask a question for the defendant which is of kind ironic.

The first task we have used for our evaluation was preparation of documents where possible questions can be formulated. The documents have been submitted to 25 people and around 233 questions have been formulated.

For this thesis work, we have collected over 15600 news articles from different newspapers (Ethiopian News Agency, Ethiopian Reporter, Walta Information Centre, etc.).

The first evaluation we have made is on the question classification and expected answer type determination. Both the rule based and the IR based question classification and expected answer type determination technique have been evaluated. The performance of our system is believed to be influenced by the performance of the question classification and expected answer type determination techniques.

The other question set we have used is the one we have collected from the corpus in [53]. These questions will help to know how much of the questions are answered from the corpus (evaluating the capability of the corpus, being wide or not). Once these question sets have been prepared, the actual evaluations have been done based on the questions. The evaluation includes, how much of the

questions are correctly answered, how much of the questions receive wrong answer, and No answer as well as whether the retrieved document bears an answer or not.

First we have to evaluate the performance of our system with the documents that have been given for different people for formulating proper questions (sample documents). What we have done is that we have first checked the performance of the system with those documents. Next we have indexed those documents with our corpus and evaluated the performance of our system.

Performance is measured by answer accuracy (precision), i.e., the fraction of the questions that were answered correctly by our system.

7.1 TESTING ENVIRONMENT

For this research work, we have used the Lucene API as a main component for information retrieval. The work in [15] has been modified to satisfy our work.

The algorithms we have developed are implemented using the Java programming language. The standard Java libraries such as Hashmap, ArrayList, StringBuffer, StreamReader, etc. have been used for text processing. We have used a number of external Java libraries for additional text processing. The eclipse Java editor has been used to develop our system. The main Java class files developed are shown in Appendix D.

We have used Windows XP Professional Edition (SP3) as an operating system. The hardware component comprises of 2 CPU of 2.00 GHz, 2.5 GB memory, and 100 GB hard disk.

7.2 QUESTION SET PREPARATION

In every question answering system, question set preparation is the main task, which is the main evaluation requirement. QA in other language, especially English, benefits from different types of questions available online in huge amount and the main task is to identify the proper question sets. TREC also have question set database where researchers used it for evaluating their system. A state-of-the-art system should pass the main criteria set along the question sets.

For our system, we have collected a number of questions from [53], the web and questionnaires distributed to 25 people. The questionnaire is distributed to Art, Engineering, Informatics, and Science students as well as three civil servant workers. Out of 233 questions collected, 219 (94%) of the questions were factoid while the remaining are list, definition, and non proper name questions. A

sample questionnaire is attached in Appendix B. A total of around 1200 questions have been prepared to evaluate our system.

7.3 EVALUATION CRITERIA

Evaluation for QA system mainly focuses on the accuracy of the answers returned. The accuracy of an answer will be evaluated in different dimensions. First of all, the length of the answer string will be evaluated. Some QA systems accept a paragraph as a correct answer, while others accept sentences to be considered as correct answer. The recent TREC QA track requires the correct answer to be an exact answer string, not a paragraph or sentence. For this research work, correct answers are piece of strings which are person names, place names, dates and numeric. Full and short forms of names are equally considered correct. For example **አትሌት አበበ ቢቂላ**, **አበበ ቢቂላ**, and **አበበ** are all considered correct. A similar rule holds for place names, dates, and numeric answers.

Evaluation of ***ተጠየቅ*** is mainly for accuracy, i.e., correctness of answers. Precision is calculated as the number of correctly answered documents over the total list of answers (correct, wrong, and No Answer). The recall is also calculated as number of correctly answered questions among the list of expected answer sets where documents will be first analyzed for the presence of correct answers. Percentage computation is done for correct answers, wrong answers, and No answers over the total answers which is the main evaluation criteria for many QA systems. In addition to precision, recall, and percentage, mean reciprocal rank (MRR) is also computed to evaluate the average rank of answers; where rank is from top one to top five. Top one means that ***ተጠየቅ*** has returned the correct answer at the top answer, and top five means that the correct answer is at position five where all answers from position one to four are wrong. Hence:

$$\text{Precision} = \frac{correct\ answers}{correct\ answers + wrong\ answers + No\ answers}$$

$$\text{Recall} = \frac{correct\ answers}{correct\ answers + missed\ answers}$$

$$\text{Percentage} = \frac{correct\ answers}{total\ answers} OR \frac{wrong\ answers}{total\ answers} OR \frac{No\ answers}{total\ answers}$$

$$\text{MRR} = \frac{\sum_{i}^{n} \frac{1}{Ri}}{n}$$

Where Ri is the rank of a given answer which ranges from 1 to 5, and n is the total number of answers (correct +wrong + No answer).

7.4 DOCUMENT NORMALIZATION AND PERFORMANCE

The performance of ***ተጠየቅ*** has been evaluated before and after document normalization. This evaluation shows us the significant effect of document normalization for performance. Consider figures 7.1 and 7.2 to see the impact of document normalization.

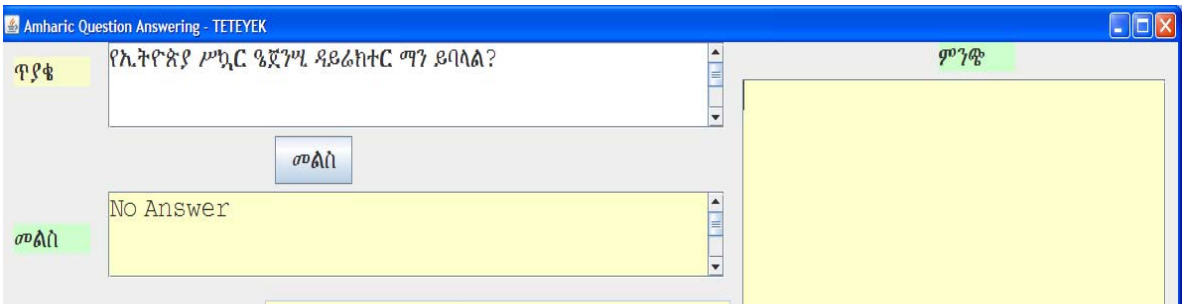

Figure 7.1: Screenshot of No answer before document normalization

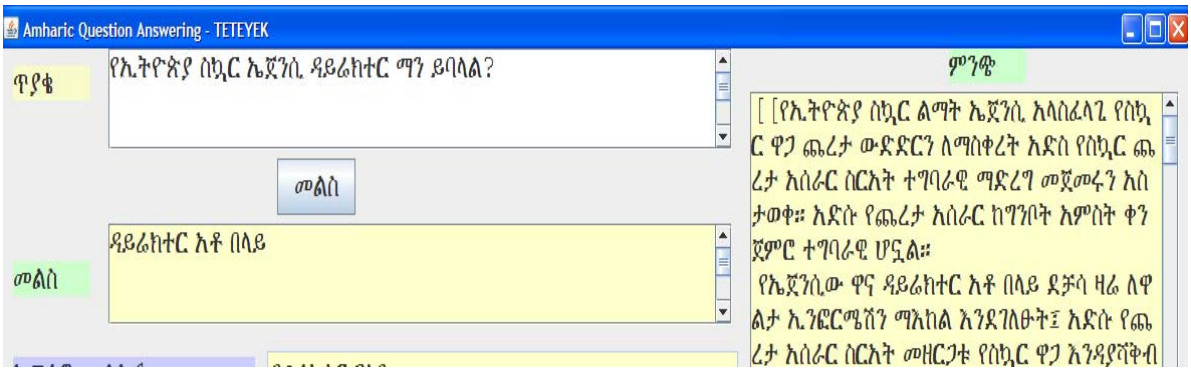

Figure 7.2: Screenshot of correct answer after document normalization

Figures 7.1 and 7.2 show that the question remains unmatched due to different character representations in the question (***ሢ*** and ***ሥ***) and in the document. The evaluation before and after document normalization for a total of 234 questions is shown in table 7.1.

Table 7.1: Character Normalization evaluation

Document	Before normalization		After normalization	
	Precision	Recall	Precision	Recall
Sentence	0.533	0.603	0.666	0.824
Paragraph	0.554	0.631	0.637	0.806
File	0.514	0.639	0.553	0.756

We can conclude from table 7.1 that document normalization has a significant effect both on precision and recall of ***ተጠየቅ***. During the experimentation, we have observed the following issues related to document normalization

- `Some questions happen to match wrong documents that results in wrong answers or No answer.
- The questions from questionnaire respondents are prepared by reading the document so that respondents follow the same writing style in the document for question preparation. The result in table 7.1 might be totally different, that is, a lesser precision and recall would be gained, if the questions had been prepared arbitrarily.
- Most questions with little variation of characters result in No answer while the document is already present in the corpus, and then recall is highly penalized. The decreased value in precision is due to variations of characters in the document that affect best answer computation.

7.5 QUESTION CLASSIFICATION EVALUATION

ተጠየቅ has been also evaluated for its question classification and expected answer type determination performance. The performance of question classification is so crucial as a wrongly classified question results in wrong answer or No answer. Hence, two techniques have been evaluated for question classification and answer type determination. The first one is rule based and the second one is IR based. Table 7.2 shows the result of both techniques.

Table 7.2: Question classification and answer type determination evaluation

Technique	Correctly classified	Wrongly classified
Rule Based	447 (89.4%)	53 (10.6%)
IR based	186 (62 %)	114 (38%)

Table 7.2 shows that the rule based question classification has been more effective when compared with the IR based classification. This is because, the IR based question classification technique needs huge amount of question sets and types indexed before a new unseen question is tested for its question type and expected answer type. The 10% wrongly classified questions means that, nearly 10% of the questions will have wrong answer or No answer.

7.6 DOCUMENT RETRIEVAL EVALUATION

The document retrieval component has been evaluated based on the presence of correct answer particles on the retrieved documents. This module incorporates sentence based, paragraph based, and file based document retrieval. Table 7.3 shows these document retrieval evaluations.

Table 7.3: Document Retrieval Evaluation

Index type	Correct answer particles present	Wrong answer particles present
Sentence	465 (93 %)	35 (7%)
Paragraph	477 (95.4 %)	23 (4.6 %)
File	486 (97.2 %)	14 (2.8 %)

We can conclude from table 7.3 that file based document retrieval is more effective in retrieving relevant documents which bear an answer particle. Sentence based document retrieval is less effective in retrieving documents which have the candidate answer particles. Here the evaluation is done based on the availability of proper answer particles in the document. Besides, this evaluation considers only the top 5 documents which are believed to have the correct answer for a question. The top 5 documents are considered because documents at the top 5 will have more similarity to the question. Documents below rank 5 are considered wrong, but our answer selection module will consider the top 100 sentences, 50 paragraphs, and 25 file for answer selection processing.

7.7 ANSWER SELECTION EVALUATION

In this Section, we will evaluate the performance of ***ተጠየቅ*** towards correct answers. The evaluation is mainly on the effectiveness of the system. We have evaluated the system towards correct answer based on the sentences, paragraphs, and files retrieved by the document retrieval component. The evaluation has been done based on Named Entity Recognition (gazetteer based) and pattern based answer selection techniques. Besides, the evaluation is done on sample data corpuses we have used where question sets are collected from questionnaires and a large corpus which contains 15600 news articles. Section 7.7.1 shows the evaluation based on sentence, paragraph, and file answer selection techniques using named entity recognition (gazetteers). Section 7.7.2 shows the pattern based answer selection technique evaluations.

7.7.1 ANSWER SELECTION EVALUATION WITH NAMED ENTITY RECOGNITION

Place and person names have been collected from [53], the Web, and from [59] as discussed in Chapter 6. The list of these place names and person names makes up our gazetteer. Returned answers will be evaluated as **correct, not correct, not exact,** and **not supported by the document**. Correct answers are answers that are exact answers for the question. For questions collected from the questionnaire, the correct answers are specifically included by the respondent so that a correct answer will be known explicitly. An incorrect answer is an answer which is of the same expected answer type but is wrong. Answers can be declared as not exact if the returned answer contains the correct answer but there is more number of strings in the answer than the correct answer. If no answer can be extracted from the document, it will be judged as no answer is found or not supported by the document. For our work, we have considered **not exact** answers as an exact answer and we have three classes; that are correct answer, wrong answer, and no answer. Figures 7.3-7.6 show correct answer, wrong answer and no answer examples respectively for different questions.

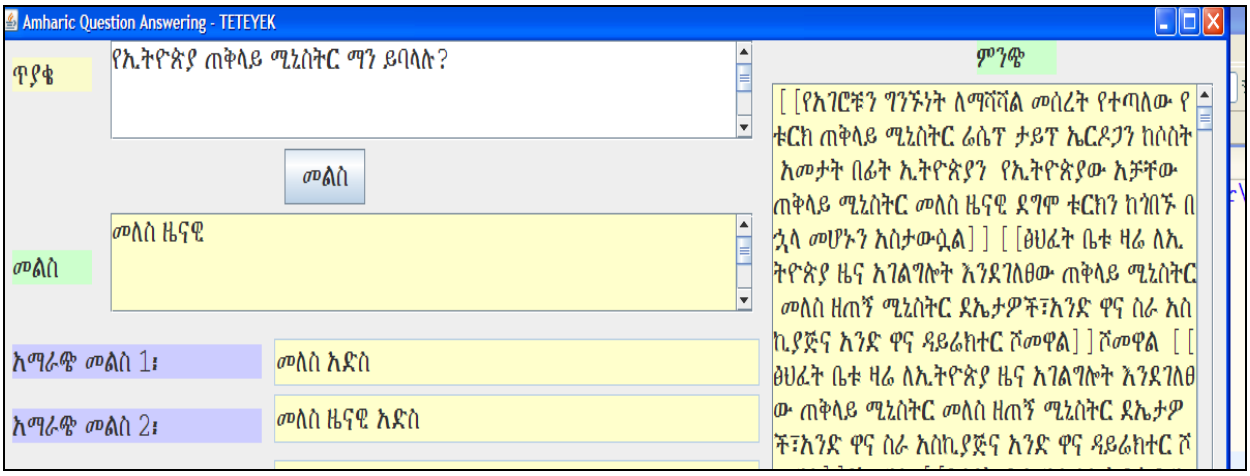

Figure 7.3: Screenshot of Correct Answer Example

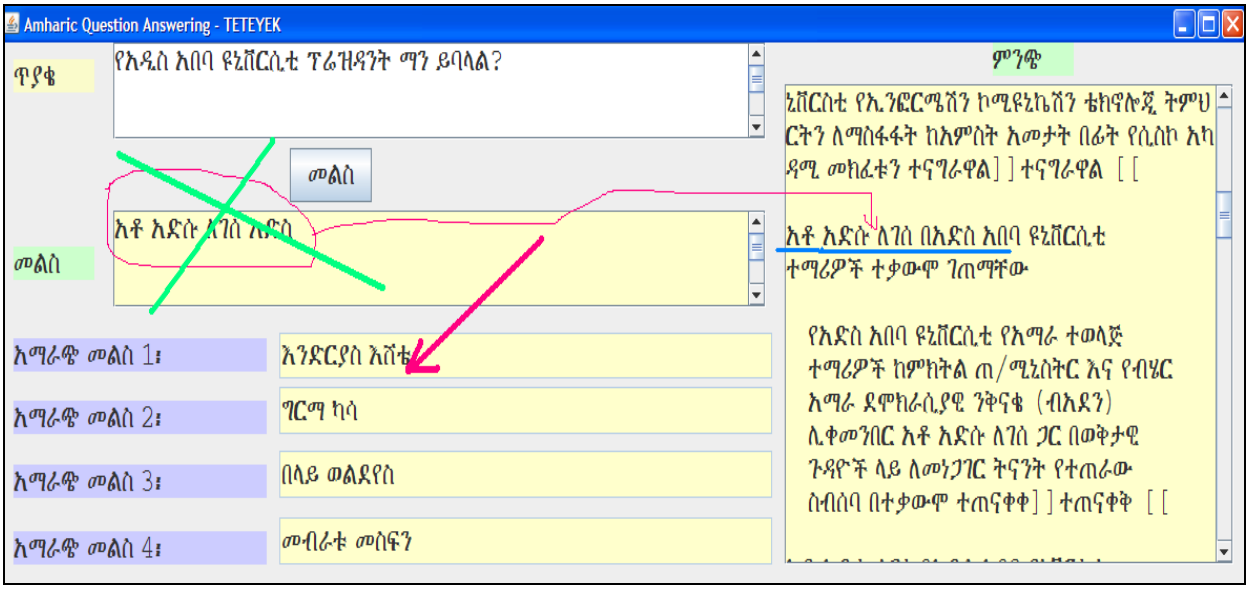

Figure 7.4: Screenshot of Correct answer at the second place

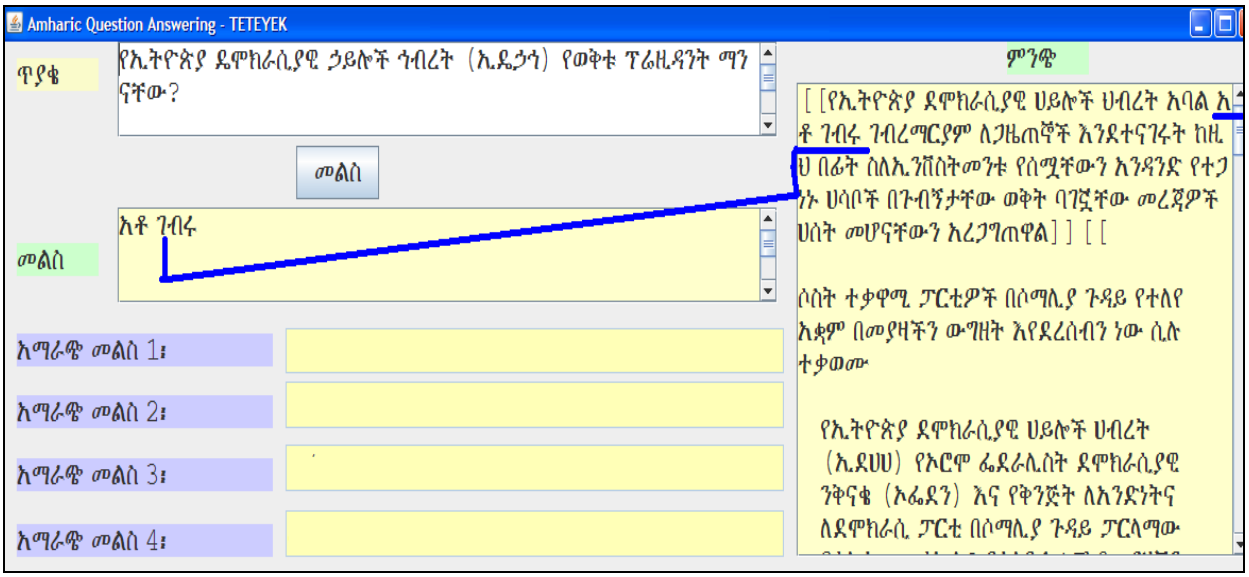

Figure 7.5: Screenshot of Wrong Answer

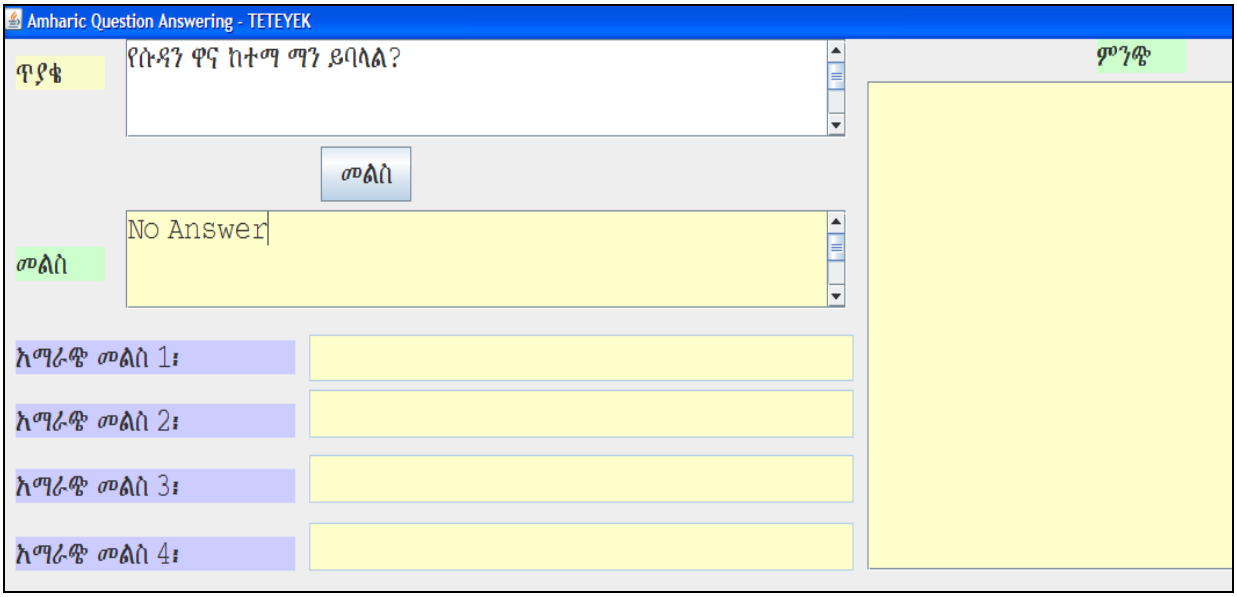

Figure 7.6: Screenshot of No answer

For person question types, we have experimented with 106 question sets and the result based on sentence, paragraph, and file based answer selection technique is shown in table 7.4.

Table 7.4: Gazetteer based answer selection for sentence, paragraph, and file documents

Document	Number of correct answers	Number of wrong answers	Number of No Answers	Missed Answers	Precision	recall	MRR
Sentence	60 (56.6 %)	30(28.3 %)	16 (15.1%)	11	0.566	0.845	0.493
Paragraph	72 (67.9%)	20 (18.9%)	14 (13.2%)	8	0.679	0.900	0.575
file	60 (56.6%)	34 (32.1 %)	12 (13.3%)	6	0.566	0.909	0.438

Table 7.4 shows that paragraph based answer selection outperforms the sentence based answer selection technique. This is due to the nature of concepts distributed in more than one sentence. The experiment shows that some questions can only be answered by the sentence based answer selection technique. This is because of the number of candidate answers in a sentence is limited and can easily be extracted. Some questions can only be answered using file based answer selection technique as the answer particle is very far to be caught by the sentence and paragraph based answer selection techniques. The precision and recall results are also encouraging. The precision result shows the system's ability in returning correct answers. The recall result shows that our system tries to identify most relevant documents (which have the expected answer). The MRR value is mainly related to the precision of the system. It is less than the precision value because we have considered answers in the

second, third, fourth and fifth positions as correct when calculating precision. In general, the MRR result shows that most of the correct answers are found at rank one. A sample of questions with answer statistics is attached in Appendix C.

Similarly, we have made evaluation for some questions to be tested using the news corpus. The result for sentence based and paragraph based answer selection techniques are shown in Table 7.5.

Table 7.5: Gazetteer based answer selection for sentence and paragraph on large corpus

Document	Correct answers	Wrong answers	No Answer	Missed answers	Precision	Recall
Sentence	36 (60 %)	23 (38.3%)	1 (1.7%)	1	0.600	0.972
Paragraph	47 (78.3%)	12 (21.7%)	0 (0%)	0	0.783	1

Table 7.5 shows that, still paragraph based answer selection is better than sentence based answer selection technique. However, the performance of the system indicates that as the number of documents increases, the probability of having a correct answer also increases. This is because, as we have more documents, document repetition will happen that means the correct answer repeats itself in more than one document. The other observation we made is that the number of No answer declines as the number of documents increases. The larger the number of documents, the less the probability of questions remained un-answered. Unfortunately, the response time increases three times which means user satisfaction will be negatively affected. We didn't check file based answer selection technique as it takes considerable amount of time for answer selection, but a similar conclusion can be made with the sentence and paragraph based answer selection techniques.

7.7.2 ANSWER SELECTION EVALUATION WITH PATTERN MATCHING

Gazetteer based answer selection is not helpful in selecting date and numeric question types. Similarly, the gazetteer we have developed for place and person name is not all inclusive so that greater number of questions remain un-answered if the gazetteer developed happened not to contain the expected answer particles. The efficiency of the system is also affected in matching every term of a document with the gazetteer content. Having larger gazetteer content means taking more amount of processing time. The pattern for numbers and dates are so wide that a multiple candidate answer might be returned for a question. The problem with the pattern based answer selection is that large number of categories should be developed to exactly answer a question. Otherwise larger candidate answers will be returned

where the answer selection module might return the wrong answer. Figures 7.7-7.10 show some examples of pattern based answer selections.

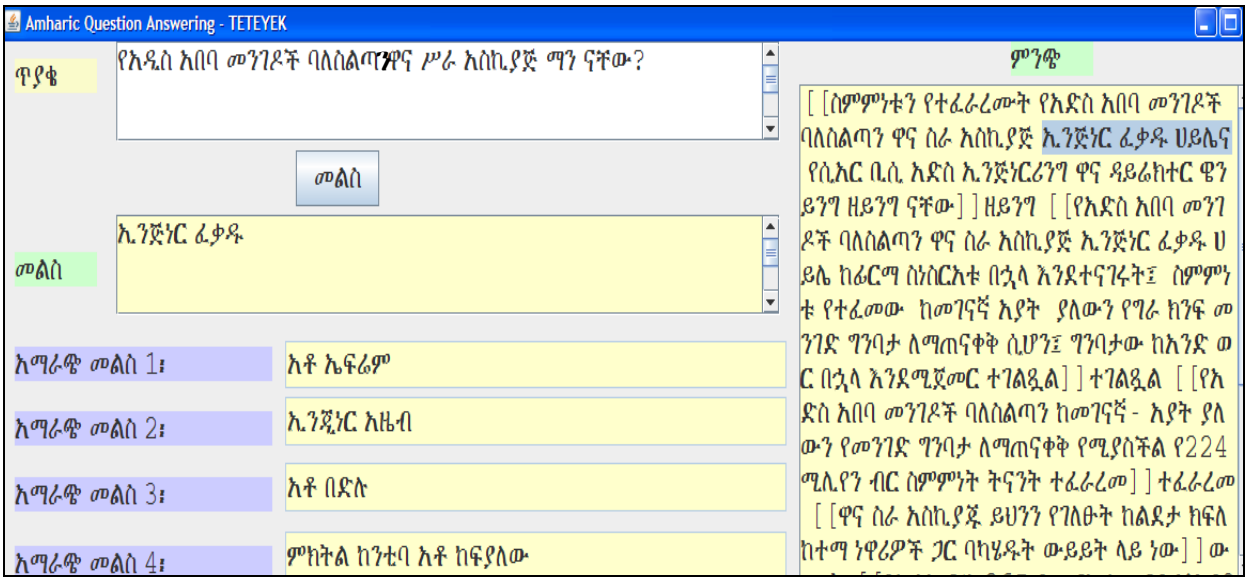

Figure 7.7: Screenshot of pattern based person name answer selection:-correct

Figure 7.7 shows that all candidate answers start with a title. This way, all possible candidate answers will be extracted without considering a gazetteer. The problem here is that sometimes person names might come without a proper title as well as terms after a title might not be person names. Consider figure 7.8:

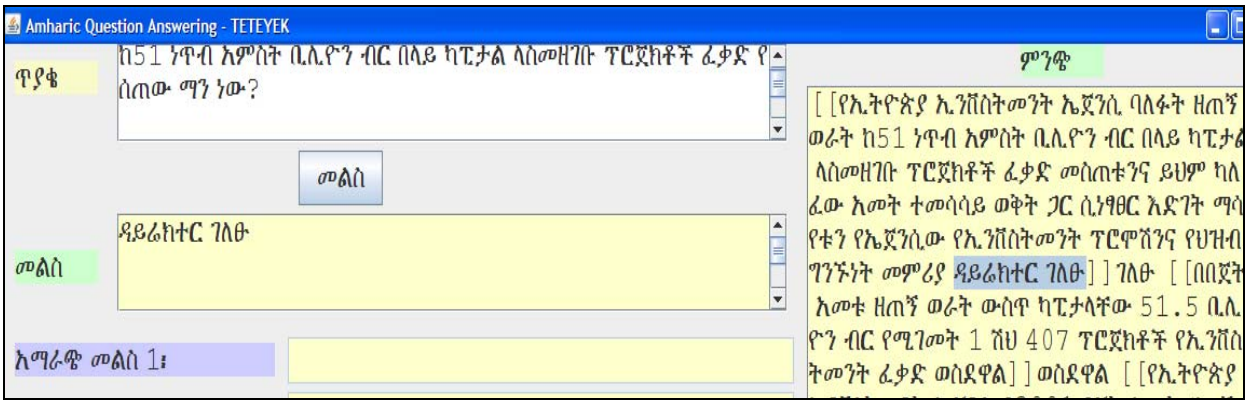

Figure 7.8: Screenshot of pattern based answer selection:-wrong, verb after title

Numeric questions behave differently when pattern based answer selection technique is used. As per our document investigation, answers for numeric questions mostly occur in a sentence boundary. The other observation we have made for numeric question types is that most questions bear an answer (exact or wrong) with minimum No answer options. This is due to the presence of numeric answer

particles in most documents, which might be correct or wrong. Figure 7.9 shows an example of numeric question answer selections.

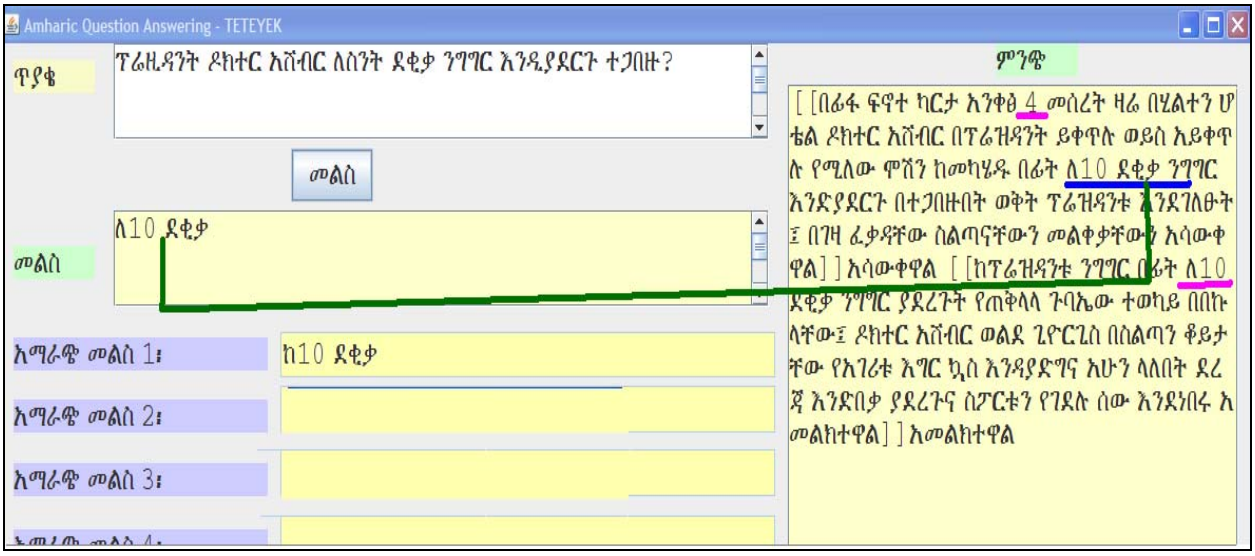

Figure 7.9: Screenshot of Pattern based numeric answer selection:- correct

The problem we have confronted in numeric answer selection technique is that a document might have multiple answer particles and the algorithm returns wrong answer based on the distance it has from the query terms. Similarly, some numeric patterns also have the same pattern as date answer selection pattern so that a date answer particle will be returned. Also the patterns we have developed do not have detailed hierarchy so that questions wrongly classified will return the wrong answer. Consider figure 7.10 for such type of errors in numeric answer selection.

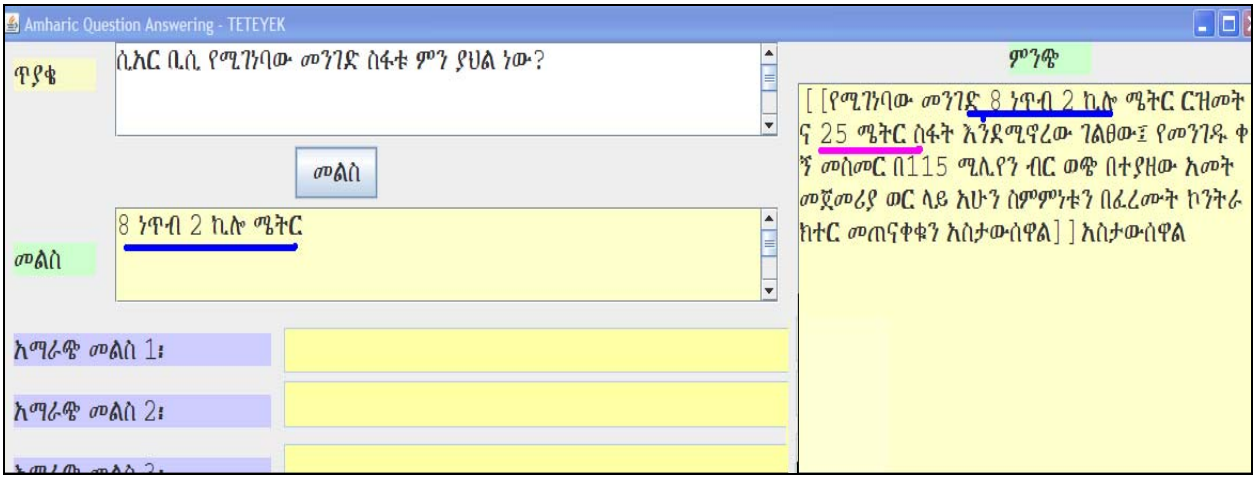

Figure 7.10 Screenshot of pattern based numeric question answer selection:-wrong

For the pattern based evaluation, we have considered 106 question sets for person name and 128 numeric questions, and the evaluation is shown in table 7.6.

Table 7.6: Pattern based answer selection evaluation

Document	Answer type	Correct answer	Wrong answer	No answer	Missed answers	precision	Recall	MRR
Sentence	Numeric	106(82.8)	18(14.1%)	4(3.1%)	13	0.828	0.891	0.710
	Person	64(60.6%)	22(20.6%)	20(18.8%)	11	0.603	0.845	0.501
paragraph	Numeric	78(60.9%)	50(39.1%)	0(0%)	27	0.609	0.742	0.522
	Person	66(62.3%)	26(24.5%)	14(13.2%)	8	0.622	0.900	0.565
file	Numeric	70(54.7%)	56(43.8%)	2(1.7%)	31	0.546	0.693	0.449
	Person	58(54.7%)	34(32.1%)	14(13.2%)	6	0.547	0.909	0.429

Table 7.6 clearly shows that paragraph based answer selection technique outperforms the other for person question types, just like the gazetteer based answer selection technique. But generally, the pattern based answer selection technique is better than the gazetteer based selection due to 1) pattern based answer selection covers wider answer matching as it doesn't depend on the list of person names in a gazetteer 2) gazetteer based answer selection matches incorrect person names that can be used for different entities such as place, entity, or even verb. There is also some efficiency gain as the pattern based answer selection technique will not consider external person name list to match an exact answer. It can also be clearly seen that sentence based numeric answer selection outperforms the others. This is due to the nature of factual Amharic documents that always appear with their numeric answer particles for the corresponding questions. Similarly, No answers are very less as there will be more numeric contents in a document matched. The wrong answers experienced in this evaluation are due to the superfluous number of candidate answers in a document.

7.8 DISCUSSION

During our experimentation, we were confronted with some problems. The first problem that we faced was that gazetteers didn't include all person names so that the system returns No answer while the document actually contains the exact answer. Besides, since person names are used as place names, entity names, and verbs, wrong terms will be returned as candidate answers. Hence, our system will rank some wrong answers higher than the actual correct answers. The pattern based person name answer selection technique experienced problems like matching no person name after a given title. For numeric and date answer selection, our system faces a problem of matching dates for numeric

questions and vice versa. For example the date 12/12/2001 can be matched both by date pattern, i.e., 12/12/2001 as is and it can also be matched with the numeric patters as 12 and 2001.

Most of the wrong answers are returned due to the problem that occurred in the question classification module. Questions of type numeric are determined as date and the system undoubtedly returns a wrong answer. The other problem we have confronted is that the stemmer we have used is so inefficient that a wrong answer will be returned. The stemmer we have used for searching and indexing have been used for answer processing too. Therefore, a question stemmed wrongly returns the wrong answer by matching the query term to the wrong document. Figure 7.11 clearly shows the impact of the stemmer for answer processing.

1. ብሮች (handful of ETBs) Stemmed to ብር
   ብራምች(continuous sunny days) Stemmed to ብር
2. በሬምች(Oxen) Stemmed to በር
   በሮች(doors) Stemmed to በር
3. ፈተናምች(exams) Stemmed to ፈት
   ፈቶች(divorcees) Stemmed to ፈት
4. ይፈተናሉ(will be examined) Stemmed to ይፈተናል
   ተፈተኑ (They took the exam) Stemmed to ተፈተን
5. ፈትሿል (he has checked) Stemed to ፈትሿል

Figure 7.11: Stemmer problem

As it can be seen from figure 7.11, ብሮች and ብራምች are stemmed to the same root word ብር that means irrelevant document will be retrieved where wrong answer or No answer will be delivered. Similarly, ይፈተናሉ and ተፈተኑ are stemmed to ይፈተናል and ተፈተን respectively while they are expected to be of same root (**መፈተን**). This means that documents that are relevant will not be matched and a correct answer is escaped.

The other serious problem we have faced is a spelling error. Most of the newspaper articles have enormous errors that lead to matching wrong answer for a given question. Similar to the spelling error is the use of different characters (fidels) for a given word (writing style). It is not a spelling error to be considered as a spelling error problem. Consider the following example:

የኢትዮጵያ ፕሬዝዳንት ማን ይባላሉ? የኢትዮጵያ ፕሬዚደንት ማን ይባላሉ?

As it can be seen from the example, **ፕሬዝዳንት** is written in two forms (**ፕሬዝዳንት** and **ፕሬዚደንት**) where both are considered correct. Similarly, the word million is written as **ሚልየን**, **ምልዮን**, **ሚሊየን**, **ሚሊዮን**, etc. Hence, exact document matching remains a problem.

The other problem we have faced in selecting best answer sentence is coreferencing. This is especially an intrinsic problem to person and place names. The name of a person will be mentioned in one sentence and the actual question may be matched to another sentence where only the coreference (such as the pronoun) of that name is mentioned. Consider the following Example:

> ..... **የውሀ አቅርቦት ማስፋፋት ስራ እያከናወነ መሆኑን የውሀ ማእድንና ኢነርጂ ቢሮ ሀላፊ አስታወቁ። ሀላፊው ዛሬ እንደገለጡት ግሊመር አፍ ሆፕ ፋውንደሽን የተባለው የፈረንሳይ ግብረ ሰናይ ድርጅት ..... አቶ አብዱልቃድር መሀመድ በተጨማሪም.....**

Here, if the user asks a question "**የማእድንና ኢነርጂ ቢሮ ሀላፊ ማን ይባላሉ**?", then, without coreference consideration, the correct answer can't be replied, especially with the sentence based answer selection technique.

Questions collected from the questionnaire have a problem of expressing the same word with the other (synonym). When the question is raised, it most of the time matches the wrong document, mostly due to stemming problem.

Lastly, during the experiment, we have discovered the following additional observations.

- Same document indexed in a sentence based, paragraph based and file based approach has different matching score even though it has the same content. This is because the Lucene API favours shorter documents. Hence sentence based document retrieval outperforms the others in relevant document retrieval in such cases.

- Questions with higher number of terms have higher probability of correct answer matching. This is because documents with more number of query terms will be favoured compared with less number of terms. Hence, questions specified in more number of terms will have higher answer probability (similar to questions in exams that are explained clearly in more number of terms will lead the student to understand the question).

- The more relevant documents repeated throughout the corpus means that returning an exact answer will be more possible.

- Most of the wrong answers are due to wrong question classification, especially for date and numeric questions, as the pattern for some numeric answer particles and date answer particles are similar.

- Lack of semantic analysis integration into our system results in a surprising answer for some questions. For example for the question "**መለስ ዜናዊ ስንት ነው?** (one of a question a student raises during experimentation) " , returns an answer "**2002**" from the document "**ጠቅላይ ሚኒስትር መለስ ዜናዊ ከ 2002 ምርጫ በኋላ ስልጣን እንደሚለቁ ገለጹ።**".

CHAPTER EIGHT

CONCLUSION AND FUTURE WORK

Information retrieval techniques have proven quite successful in locating relevant documents. The IR systems retrieve a number of ranked documents that users should go through to get the pertinent information. It has been criticized in failing to bring the exact response the user is in need. In most search engines, the IR component retrieves a number of documents that may have duplicates; but it is the user who will judge such things. Also in IR systems, the user is required to formulate a proper query to maximize relevant document retrieval. For inexperienced Internet users, getting the required information will be more difficult using IR systems. Moreover, users need a piece of factual information which will be located somewhere in a document where users are required to read all the documents.

As a result of the huge information overload, information extraction becomes the focus of many researchers. Information extraction is the identification of instances of a particular class of events or relationships in a natural language text, and the extraction of the relevant ones. Hence, users will not be flooded with huge information, rather specific information (in the form of text, or sentence, or paragraph) will be returned. The document will be further chopped down in to pieces of factual information where that piece of information by itself is meaningful and capable of representing the document.

Question answering is founded on information extraction where a given fact is to be returned for the user. In QA, users will pose their question in natural language and the system will return an exact and precise answer.

8.1 CONCLUSIONS

In this thesis we have developed a QA system for Amharic that has different components. The question processing module will classify the question to appropriate question types, determine the expected answer type, and generate proper IR query. The document retrieval component will retrieve relevant documents that will be further processed by the later modules. The sentence/paragraph ranker will re-rank sentences and paragraphs based on the answer particle in them. The final module, the answer selection module, will select the best answers to the user with an additional source for the user in case detailed information is needed.

Chapter Eight: Conclusion and Future Work

This research work attempted to identify the basic language specific issues in question answering. The first task we have tackled is normalizing the document so that a standard document will be indexed and matching relevant documents during searching will be maximized. We have also identified proper question particles as well as question focuses that will help in classifying the question. Gazetteer based and pattern based answer selection algorithms have been developed to maximize correct answer selection. Our algorithm first identifies all possible answer particles in a document. Once the answer particles are identified, the distance of every question particle with the question terms will be calculated. The one with the minimum distance from the query terms will be considered the best candidate answer of that document. Once candidate answers are selected from every document, candidate answers which have been repeated more than once (i.e., appeared in more than one document) will be given higher rank. Candidate answers with maximum number of query terms matched in a document will be given higher priority in case a similar rank is given for two or more candidate answers.

The evaluation of our system, being the first Amharic QA system, shows promising performance. The rule based question classification module classifies about 89% of the question correctly. The document retrieval component shows greater coverage of relevant document retrieval (97%) while the sentence based retrieval has the least (93%) which contributes to the better recall of our system. The gazetteer based answer selection using a paragraph answer selection technique answers 72% of the questions correctly which can be considered as promising. The file based answer selection technique exhibits better recall (0.909) which indicates that most relevant documents which are thought to have the correct answer are returned. The pattern based answer selection technique has better accuracy for person names using paragraph based answer selection technique while the sentence based answer selection technique has outperformed in numeric and date question types. In general, our algorithms and tools have shown good performance compared with high-resourced language QA systems such as English.

8.2 Contribution of the Work

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

- ✓ The study has adopted the efforts made towards English QA systems techniques to Amharic.
- ✓ The study has paved the way to identify language dependent components specific to Amharic question answering.

- ✓ The study identified key components of Amharic QA systems which can be considered a framework for factoid questions.
- ✓ The study showed the strategy, algorithms, and techniques in developing Amharic QA system.
- ✓ The study showed how questions in Amharic can be classified hierarchically (coarse and fine grained based), what are the specific question focuses for different questions, and the function of question particles to determine question type and expected answer types.
- ✓ This study also showed how information extraction can be accomplished in Amharic based on the standard off-the-shelf information retrieval techniques available.
- ✓ The study identified basic challenges in developing Amharic QA systems and the possible strategies to solve those challenges.

8.3 FUTURE WORK

Question answering is a very complex task, which consumes more time, and needs a number of different NLP tools. Hence, there are a number of rooms for improvement and modification for Amharic question answering. Below are some of the recommendations we propose for future work.

- Developing automatic named entity recognizer: The gazetteer we have used has limitations such as usage of a single named entity for multiple entities (such as person and place). Developing an automatic named entity recognizer will help the QA system to automatically detect the expected answer.
- Incorporating a parser and part of speech tagger: The NER will detect named entities in a document. A sentence parser will further help the QA system to know the structure of the question and the expected answer sentence. Besides, there is no POS tagger available publicly to integrate with our QA system. Integrating POS tagger will help the answer processing component of the QA system so that wrong answer particles, such as considering a verb as proper noun, will be eliminated.
- Developing Amharic WordNet: Word synonym, hyponym, antonym, metonym, meronym and so on help to match wider number of relevant documents. By using Amharic synonyms and the like, we believe that Amharic WordNet is very beneficial.

- Enhancing the Amharic stemmer: The stemmer that we have used brought some drawbacks both for document retrieval and answer processing algorithms. It will be better to develop a state-of-the-art stemmer which we believe will bring a significant change to the performance of QA systems.
- Incorporating Machine learning and statistical Question classifications: the rule based and IR based question classifications have some limitations. The rule based approach does not include all possible patterns of questions and the IR approach also does not help as the number of questions and question types indexed are very small. The machine learning and statistical approaches show better performance for other QA systems such as English [60] and we hope it will also help for Amharic QA systems as well.
- Integrating with other search engines: for this research work, documents have been collected manually with the help of third party tools such as **DownThemAll** of Firefox and **WinHTTrack website copier**[*]. It will be better to incorporate a crawler component which will interact with the main search engines (Google, Yahoo, etc.) and Amharic Websites for collecting relevant documents.
- Extending to other question types: This research work shows that, even with minimal NLP tools, it would be possible to handle other question types such as list, define, and so on. Extending this work to other question types will be beneficial for wider applications where only a piece of information is not sought.
- Incorporating Amharic spell checker: most of the wrong answers and wrong documents returned are due to spelling errors. Incorporating spell checker will enhance the performance of our system.
- Implementing for specific applications: The QA system can be easily implemented to satisfy the needs of some organizations for specific projects. It can be developed for customer service support such as e-commerce and e-governance.

[*] HTTrack is a free (GPL, libre/free software) and easy-to-use offline browser utility, http://www.httrack.com/

REFERECES

[1] Qinglin Guo, Kehe Wu, Wei Li, 2007. The Research and Realization about Question Answer System based on Natural Language Processing, Proceedings of the Second International Conference on Innovative Computing, Information and Control.

[2] Hu, H. Jiang, P. Ren, F. Kuroiwa, S. 2005. Web-based Question Answering System for Restricted Domain Based of Integrating Method Using Semantic Information Natural Language Processing and Knowledge Engineering, 2005. IEEE NLP-KE '05. Proceedings of 2005 IEEE International Conference.

[3] Anne-Laure Ligozat, Brigitte Grau, Anne Vilnat, Isabelle Robba, Arnaud Grappy, 2007. Towards an automatic validation of answers in Question Answering, LIMSI-CNRS 91403 Orsay CEDEX France.

[4] http://en.wikipedia.org/wiki/Question_answering, last accessed on 04 September 2008

[5] http://trec.nist.gov/tracks.html, last accessed on 08 September 2008.

[6] Dongfeng Cai Yanju Dong Dexin Lv Guiping Zhang Xuelei Miao, 2004. A web based Chinese Question Answering System, Natural Language Processing and Knowledge Engineering, 2005. IEEE NLP-KE '05. Proceedings of 2005 IEEE International Conference.

[7] Jiangping Chen, Anne R. Diekema, Mary D. Taffet, Nancy McCracken, Necati Ercan Ozgencil, Ozgur Yilmazel, and Elizabeth D. Liddy, 2002. Question Answering :CNLP at the TREC-10 Question Answering Track, In Proceedings of the Tenth Text Retrieval Conference (TREC).

[9] Jignashu Parikh, M. Narasimha Murty, 2002. Adapting Question Answering Techniques to the Web, Proceedings of the Language Engineering Conference (LEC'02).

[10] Dongfeng Cai, Yu Bai, Yanju Dong, Lei Liu, 2007. Chinese Question Classification Using Combination Approach, Proceedings of the Third International Conference on Semantics, Knowledge and Grid.

[11] Shouning Qu, Bing Zhang , Xinsheng Yu , Qin Wang, 2008. The Development and Application of Chinese Intelligent Question Answering System Based on J2EE Technology, Proceedings of the 1st international conference on Forensic applications and techniques in telecommunications, information, and multimedia and workshop.

[12] Zheng-Tao Yu Yan-Xia Qiu Jin-Hui Deng Lu Han Cun-Li Mao Xiang-Yan Meng , 2007, Research on Chinese FAQ questions Answering System in Restricted Domain, Machine Learning and Cybernetics, 2007 International Conference.

[13] Sameer S. Pradhan, Valerie Krugler, Wayne Ward, Dan Jurafsky and James H. Martin, Using Semantic Representations in Question Answering, Center for Spoken Language Research University of Colorado Boulder, CO 80309-0594, USA.

[14] Tomek Strzalkowski and Sanda Harabagiu, 2008, Advances in Open Domain Question Answering, Published by Springer, ISBN 978-1-4020-4746-6, The Netherlands.

[15] Tessema Mindaye, 2007, design and implementation of Amharic Search Engine, A Thesis Submitted to the School of Graduate Studies of the Addis Ababa University in partial fulfillment for the Degree of Master of Science in Computer Science.

[16] Steven Bird, Ewan Klein, Edward Loper, Sep 10, 2008, Natural Language Processing, Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 United States, Draft Book.

[17] http://en.wikipedia.org/wiki/Natural_language_processing, last accessed on October 01, 2008.

[18] http://www.omniglot.com/writing/amharic.htm, last accessed on October 01, 2008

[19] http://en.wikipedia.org/wiki/Amharic_language, last accessed on October 1, 2008

[20] http://trec.nist.gov/data/qa/2007_qadata/QA2007_testset.xml.txt, last accessed on October 03, 2008

[21] http://trec.nist.gov/data/qa/2007_qadata/factoid_judgments.txt, last accessed on October 03, 2008

[22] **ጌታሁን አማረ፣** 1989 **የአማርኛ ሰዋሰው በቀላል አቀራረብ**

[23] Mark A. Greenwood, 2005, AnswerFinder: Question Answering from your Desktop, Department of Computer Science University of Sheffield Regent Court, Portobello Road Sheffield S1 4DP UK

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

[25] Christof Monz, 2003, Document Retrieval in the Context of Question Answering, In Proceedings of the 25th European Conference on Information Retrieval Research (ECIR-03.

[26] Matthew W. Bilotti, Boris Katz, and Jimmy Lin, 2004, What Works Better for Question Answering: Stemming or Morphological Query Expansion?, Massachusetts Institute of TechnologyCambridge, Massachusetts, USA.

[27] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, Chin-Yew Lin., Question Answering in Webclopedia, In Proceedings of the Ninth Text REtrieval Conference (TREC-9.

[28] Thomas Morton, 2005, Using Semantic Relations to Improve Information Retrieval, PhD Dissertation in Computer and Information Science.

[29] Eduard Hovy , Ulf Hermjakob , Deepak Ravichandran, 2002, A Question/Answer Typology with Surface Text Patterns, Proceedings of the second international conference on Human Language Technology Research.

[30] Charles L. A. Clarke, Egidio L. Terra, 2003, Passage Retrieval vs. Document Retrieval for Factoid Question Answering, Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval.

[31] Steven Abney, Michael Collins, Amit Singhal, 2000, Answer Extraction, Proceedings of the sixth conference on Applied natural language processing.

[32] Bassam Hammo, Hani Abu-Salem, Steven Lytinen , DePaul University, 2002, QARAB: A Question Answering System to Support the Arabic Language, DePaul University School of Computer Science, Telecommunications and Information Systems 243 S. Wabash Avenue, Chicago IL 60604.

[33] http://www.lonweb.org/link-amharic.htm, last accessed on March 30, 2009.

[34] Leslau, Wolf, 1969, An Amharic Reference Grammar, California University, Los Angeles, Dept. of Near Eastern and African Languages.

[35] Mengistu Amberber, Helen De Hoop, 2005, Competition and Variation in Natural Languages the case for case, Linguistics Department, The University of New South Wales, Sydney, Australia.

[36] http://jrgraphix.net/research/unicode_blocks.php?block=31, last accessed on March 31, 2009.

[37] http://www.englishclub.com/vocabulary/wh-question-words.htm, last accessed on March 31, 2009.

[38] Xiaoyan Li and W. Bruce Croft, 2002, Evaluating Question-Answering Techniques in Chinese, In NIST Special Publication: The 10 th Text Retrieval Conference.

[39] Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze, 2009, An Introduction To Information Retrieval, Draft Book.

[40] Charles L.A. Clarke, Gordon V. Cormack, Thomas R. Lynam, 2001, Exploiting Redundancy in Question Answering, Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval.

[41] INQUERY Query Help, http://ciir.cs.umass.edu/irdemo/inqinfo/ inqueryhelp.html, last accessed on April 4, 2009.

[42] Richard F. E. Sutcliffe, Jia Xu Michael Mulcahy, 2005, Chinese Question Answering using the DLT System at NTCIR 2005, Proceedings of NTCIR-5 Workshop Meeting, December 6-9, 2005, Tokyo, Japan.

[43] Otis Gospodnetic, Erik Hatcher, 2005, Lucene in Action, Manning Publications Co., 209 Bruce Park Avenue, Greenwich, CT 06830, ISBN 1-932394-28-1.

[44] Abdessamad Echihabi, Ulf Hermjakob, Eduard Hovy, Daniel Marcu, Eric Melz, Deepack Ravichandran, 2004, How to Select an Answer String, Advances in Textual Question Answering, Tomek Strzalkowski and Sanda Harabagiu eds., Kluwer.

[45] Diego Moll´a and Menno van Zaanen and Daniel Smith, 2006, Named Entity Recognition for Question Answering, Proceedings of the 2006 Australasian Language Technology Workshop (ALTW2006).

[46] Praveen Kumar, Shrikant Kashyap, Ankush Mittal, Sumit Gupta, 2005, A Hindi Question Answering system for E-learning documents, Proceedings of the 2005 3rd International Conference on Intelligent Sensing and Information Processing.

[47] Cheng-Wei Lee, Cheng-Wei Shih, Min-Yuh Day, Tzong-Han Tsai, Tian-Jian Jiang, Chia-Wei Wu, Cheng-Lung Sung, Yu-Ren Chen, Shih-Hung Wu, Wen-Lian Hsu, 2005, ASQA: Academia Sinica Question Answering System for NTCIR-5 CLQA, Proceedings of NTCIR-5 Workshop Meeting, December 6-9, 2005, Tokyo, Japan

[48] Ian F. Darwin, June 2004, Java Cookbook, 2nd Edition, O'Reilly, ISBN: 0-596-00701-9

[49] Stefan Schlobach, Marius Olsthoorn, Maarten de Rijke, 2004, Type Checking in Open-Domain Question Answering, Informatics Institute, University of Amsterdam, Proceedings ECAI 2004, Valencia, 2004. IOS Press.

[50] Rawia Awadallah and Andreas Rauber, 2006, Web-based Multiple Choice Question Answering for English and Arabic Questions, Department of Software Technology and Interactive Systems European Colloquium on IR Research (ECIR-2006).

[51] Steven Sinha and Srini Narayanan, 2005, Model-based Answer Selection, International Computer Science Institute University of California, Berkeley 1947 Center Street, Suite 600, American Association for Artificial Intelligence 2005.

[52] http://nlp.amharic.org/resources/, last accessed on may 01, 2009.

[53] Tamesol Communication, Answer and Question corpus, from 2000-2009

[54] Kevyn CollinsThompson, Jamie Callan, Egidio Terra, Charles L.A. Clarke, 2004, The Effect of Document retrieval quality on Factoid Question Answering Performance, Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval.

[55] http://gate.ac.uk/ie/, last accessed on may o3, 2009.

[56] Lili Aunimo, 2005, A Question Typology and Feature Set for QA, Proceedings of the Workshop for Knowledge and Reasoning for Answering Questions, held in conjuction with IJCAI-05, July 2005, Edinburgh, UK.

[57] Berthier Ribeiro-Neto Ricardo Baeza-Yates, 1999, Modern Information Retrieval. Addison Wesley, ISBN-13: 978-0201398298.

[58] Andrew Hickl, Kirk Roberts, Bryan Rink, Jeremy Bensley, Tobias Jungen, Ying Shi, and John Williams, 2007, Question Answering with LCC's Chaucer-2 at TREC 200t, Proceedings of the 2007 Text Retrieval Conference (TREC 2007).

[59] http://am.ethiopiabook.com, last accessed on April 15,2009.

[60] Ulf Hermjakob, 2002, Parsing and Question Classification for Question Answering, Proceedings of the workshop on Open-domain question answering - Volume 12.

APPENDICES

Appendix A: Coarse and Fine grained Expected answer types for question classification

- የመልስ አይነቶች
  - ስም
    - ቦታ
      - አገር
        - ከተማ
        - ክ/ከተማ
        - ወረዳ
        - ቀበሌ
        - ወንዝ
        - ተራራ
        - ሀይቅ
        - ውቅያኖስ
        - ሆቴል
        - ክልል
        - ዩኒቨርሲቲ
        - ዋሻ
        - ሐውልት
        - ድልድይ
        - ደሴት
        - ት/ቤት
        - መስጅድ
        - ቤተ-ክርስቲያን
        - ሌላ
      - ፕላኔት
      - ድርጅት
    - ከለር/መልክ
    - ድረ-ገፅ
    - ሀይማኖት
    - ቋንቋ
    - ሥራ/ሙያ
    - ሰው
      - ፕረዝዳንት
      - ተጨዋች
      - ደራሲ
      - ሀላፊ
      - ቅጥያ ስም
      - ከንቲባ/አስተዳዳሪ
      - ሌላ
    - በሽታ
    - መጠጥ
    - ምግብ
    - እንስሳት
    - እቃ
      - መኪና
      - አውሮፕላን
      - ሌላ
    - ሌላ
  - ስፖርት
    - ተጨዋች
    - አትሌት
    - ስታድየም
    - ቡድን
    - አሰልጣኝ
    - አምበል
  - ሙዚቃ
    - መሣሪያ

- ስፖ
- ዘፋኝ
- አርቲስት
- ደራሲ

❖ ሳይንስ

❖ መጠን

- ጊዜ
- ርቀት

- ገንዘብ
- ኮምፒዩተር
- ፍጥነት
- መፍቀት
- ሌላ

❖ ዓመት በዓል

❖ ህገ-መንግስት

Appendix B: Sample Questionnaire to prepare question sets from a document for testing and question classification

ተጠየቅ is a question answering system for Amharic. We have given you these documents so that you can prepare factual questions (place, person, numeric, and date) that will help test the performance of **ተጠየቅ**. For example for the sentence: **የኢትዮጵያ ጠቅላይ ሚኒስትር አቶ መለስ ዜናዊ የኢትዮጵያ ህዝብ ብዛት 80 ሚሊየን እንደሚጠጋ መጋቢት 21 ቀን 2001 ዓ.ም. አዲስ አበባ በተካሄደው የአፍሪቃ ህብረት ስብሰባ ላይ ገለጹ።** We can ask the following questions:

1. **የኢትዮጵያ ጠቅላይ ሚኒስትር ማን ይባላል?**
2. **የኢትዮጵያ ህዝብ ብዛት ምን ያህል ነው?**
3. **የኢትዮጵያ ጠቅላይ ሚኒስትር ስለ ኢትዮጵያ ህዝብ ብዛት መግለጫ የሰጡት መቼ ነው?**
4. **የአፍሪቃ ህብረት ስብሰባ የት ተካሄደ?**

If you can, prepare more number of questions for each question type (place, person, number, date). The question must have an answer in the document and better include the answer for the question. Thanks for your help.

1. **ሐዋሳ ግንቦት 12/2001/ ዋኢማ/ በደቡብ ብሄር ብሄረሰቦችና ህዝቦች ክልል በተያዘው የትምህርት ዘመን 44ሺ የሚጠጉ ህፃናት በምገባ ፕሮግራም ትምህርታቸውን እየተከታተሉ መሆናቸውን የክልሉ ትምህርት ቢሮ አስታወቀ።**

   የቢሮው የትምህርት ልማት እቅድ ዝግጅት ክትትልና ግምገማ የስራ ሂደት ባለሙያ አቶ ልዑልሰገድ ይመር ዛሬ ለዋልታ ኢንፎርሜሽን ማዕከል እንዳስታወቁት፤ በክልሉ ድርቅ በተደጋጋሚ ከሚያጠቃቸው ወረዳዎች መካከል በምገባ ፕሮግራም ለታቀፉ 90 ትምህርት ቤቶች የዓለም የምግብ ፕሮግራም በመደበው ከ531 ሜትሪክ ቶን በላይ አልሚ ምግብ 44ሺ የሚጠጉ ተማሪዎች ትምህርታቸውን እየተከታተሉ ነው።

   የዓለም የምግብ ፕሮግራም በክልሉ የተለያዩ ዞኖች በሚገኙ 13 ወረዳዎች እያካሄደ ያለው ምገባ ድርቅ በሚበዛባቸው አካባቢዎች ያሉ ሕፃናት የመማር እድል እንዲያገኙ ከማስቻሉም በተጨማሪ የክልሉን ጥቅል የትምህርት ተሳትፎ ለማሳደግ የበኩሉን አስተዋጽኦ እያደረገ መሆኑን አቶ ልዑልሰገድ አስረድተዋል።

   በቢሮውና በፕሮግራሙ አስተባባሪዎች የተቀናጀ ስራም በፕሮግራሙ ለአንደኛው ወሰነ ትምህርት የተመደበው አልሚ ምግብ በአግባቡ ጥቅም ላይ እንዲውል የተደረገ መሆኑን የጠቆሙት ባለሙያው፤ የሁለተኛው ወሰነ ትምህርት የምገባ ፕሮግራሙም ተጠናክሮ መቀጠሉን አመልክተዋል።
   ፕሮግራሙ በተያዘው የትምህርት ዘመን በክልሉ የምገባ መርሃግብር በማካሄድ ትምህርታቸውን እንዲከታተሉ ካስቻላቸው 44ሺ ከሚጠጉት ህፃናት ውስጥም ከግማሽ በላይ የሚሆኑት ሴቶች መሆናቸውን ከአቶ ልዑልሰገድ ገለፃ ለመረዳት መቻሉን ዋልታ ኢንፎርሜሽን ማዕከል ዘግቧል።

ጥያቄዎች

---

2. **የጊቤ ሦስት የኤሌክትሪክ ኃይል ማመንጫ ግንባታ** 30 **በመቶ ያህሉ መጠናቀቁን ግንባታው ሙሉ ለሙሉ ሲጠናቀቅ አሁን ያለውን ከ**900 **ሜጋ ዋት የማይበልጥ የኤሌክትሪክ ኃይል አቅም በሙሉ የሚቀይረው መሆኑን ኢንጂነር አዜብ አስናቀ የፕሮጀክቱ ሥራ አስኪያጅ ገለፀ።።**

በሁለት መቶ ስኩየር ኪሎ ሜትር ላይ የሚያርፍ ሐይቅ የሚኖረው የዚሁ ኃይል ማመንጫ ግንባታ ወጪ አንድ ነጥብ አምስት ቢሊዮን ዩሮ እንደሚደርስ ከሥራ አስኪያጂ ማብራሪያ ለመረዳት ተችሏል።።

ጥያቄዎች

---

3. **የዓለም ኢኮኖሚ አሁን ባለበት ደረጃ ላይ ቢገኝም የአካውንታንቶች ተፈላጊነት በጣም ከፍ ያለ መሆኑን ወይዘሮ ሐክሚት አብደላ የአሶሴሽን ኦፍ ቻርተርድ አካውንታንትስ የኢትዮጵያ ዳይሬክተር አስታወቁ።።**
ዳይሬክተሯ ይህንን ያስታወቁት አሶሴሽኑ ለአንድ ዓመት ያሰለጠናቸው 50 **ኢትዮጵያውያን ግንቦት** 15 **ቀን** 2001 **ዓ.ም ማስመረቁን አስመልክተው ባወጡት ጋዜጣዊ መግለጫ ላይ ነው።።**

ወይዘሮ ሐክሚት በዚሁ መግለጫቸው ላይ እንደገለፁት አካውንታንቶች በንግዱ ዘርፍ እሴትን፣ ትርፍና ኪሳራን በአግባቡ በመያዝ የጐላ አስተዋፅኦ እንደሚያበረክቱ አስረድተዋል።።

ጥያቄዎች

---

4. መቀሌ ግንቦት 12/2001/ዋኢማ/ በመቀሌ ከተማ የዓዲ ሐቂ ክፍለ ከተማ ፍርድ ቤት "በባህላዊ ህክምና የታመመን አድናለሁ ፤ የተቀበረ መድሃኒትን አወጣለሁ" በማለት ህብረተሰቡን ያጭበረበረ ግለሰብ የአምስት ዓመት ጽኑ እሥራትና የገንዘብ ቅጣት ተወሰነበት።

ፍርድ ቤቱ ዛሬ ጥዋት በዋለው ችሎት ተከሳሽ ሰለሞን ታደለ አረጋይ የተባለ የክፍለ ከተማው ነዋሪ የህብረሰቡን ሰላም የሚያናጋ አደገኛ የማታለል ወንጀል መፈጸሙ በማሰረጃ በመረጋገጡ የአምስት ዓመት ጽኑ እሥራትና የአምስት ሺ ብር ቅጣት ወስኖበታል ።

የፍርድ ቤቱ ዳኛ ወይዘሮ ምሕረት ተክላይ በችሎቱ ላይ ባቀረቡት ውሳኔ እንዳብራሩት፤ ግለሰቡ በባህላዊ ህክምና ሰዎችን አድናለሁ በማለት በርካታ ገንዘብ በመቀበል፣ ባለትዳሮችን በማጣላት በሞት እንዲለያዩ እሰከ ማድረግ ደርሷል።

በተከሳሹ የባህላዊ ህክምና ፈቃድ ወረቀት ላይ የራሱ ፎቶ ግራፍ ቢኖርም መታወቂያው የሌላ ሰው ስም ሆኖ በመገኘቱ ቅጣቱ ሊከብድ መቻሉን ዳኛዋ አስረድተዋል ።

ግለሰቡ በሰጠው አስተያየት ላለፉት ሦስት ዓመታት ህጋዊ የባህላዊ ህክምና ፈቃድ አግኝቶ በህጋዊ መንገድ ህብረተሰቡን በባህላዊ ህክምናው ሲያገለግል መቆየቱን ጠቁሞ፤ የተሰጠው የፍርድ ውሳኔ አግባብ አይደለም ማለቱን ዋልታ ኢንፎርሜሽን ማዕከል ዘግቧል።

ጥያቄዎች

Appendix C: Sample Question Sets with Answer distribution Statistics: ***√=Correct Answer, WR = Wrong Answer, NA = No Answer***

	Sentence Based					Paragraph Based					File Based				
	top1	top2	top3	top4	top5	top1	top2	top3	top4	top5	top1	top2	top3	top4	top5
የኢትዮጵያ እግር ኳስ ፌዴሬሽን ፕሬዝዳንት ማን ይባላል?	√					√					√				
የአድስ አበባ መንገዶች ባለስልጣን ዋና ስራ አስኪያጅ ማን ይባላሉ?	√					√					WR	WR	WR		
የሲአር ቢሲ አድስ ኢንጅነሪንግ ዋና ዳይሬክተር ማን ይባላል?	√					√					√				
የአደራ ፊልም ዋና ገጸ ባህሪ ማን ናት?	NA	WR	WR			NA	WR				NA	WR			
ሰለ አዜ ገላውዴዎስ ሞት በግእዝ የተፃፈውን ወደ አማርኛ የተረጎመው ማን ነው?	√						√				√				
የኢትዮጵያ የአየር ትራፊክ ተቆጣጣሪዎች መህበር ያዘጋጀውን ፊልም የመረቁት ማን ናቸው?	√					√					√	NA			
የኢትዮጵያ የአየር ትራፊክ ተቆጣጣሪዎች መህበር ፕሬዝዳንት ማን ነው?		√				√						√			
በደቡብ ክልል የትምህርት ልማት እቅድ ዝግጅት ክትትልና ግምገማ የስራ ሂደት ባለሙያ ማን ይባላሉ?	√					√					√	NA			
የቶፕ ኮንስትራክሽን አመራር አባል ማን ይባላሉ?	√	NA				√					√				
የአዲስ አበባ ከተማ አስተዳደር ምክትል ከንቲባ ማን ይባላሉ?	√						√				WR	WR	WR	WR	
የድሬደዋ ከተማ ከንቲባ ማን ይባላሉ?	√					√					√				
ከመገናኛ-አያት ያለውን የመንገድ ግንባታ ለማጠናቀቅ የሚያስችል የ224 ሚሊዮን ብር ስምምነት የተፈራረመው ማን ነው?	√	WR	WR			√	WR				√	WR			
የአዲስ አበባ ዋና ሥራ አስኪያጅ ማን ናቸው?	√					√					WR				
የኢንቨስትመንት ኤጀንሲ ዳይሬክተር ማን ናቸው?	WR	WR				WR					WR	WR	WR		
የኦፌዴን ዋና ፀሀፊ ማን ናቸው?	√					√					WR	WR	WR		
የትምህርት ሚኒስቴር የህዝብ ግንኙነት ሀላፊ ማን ናቸው?	√	NA				√					√				
የአፍዴሀግ ተቃዋሚ ግንባር ሊቀመንበር ማን ናቸው?	√					√					WR				
የቶፕ ኮንስትራክሽን ሀላፊነቱ የተወሰነ የህብረት ማህበር አመራር አባል የሆኑት ማን ናቸው?	√	NA				√	NA				√	NA			
የደቡብ ክልል ይትምህርት ልማት እቅድ ዝግጅት ክትትልና ግምገማ	√		WR			√	√				√	WR	WR		

የስራ ሂደት ባለሙያ ማን ይባላሉ?															
በሶማሌ ክልል ደገሀቡር ዞን የጋሻሞ ወረዳ አስተዳዳሪ ማን ይባላሉ?	√					√					√				
የሾላ ምግብ ዝግጅት ባልትና ማህበር ሊቀመንበር ማን ይባላሉ?	√					√					√				
የጅንካ ከተማ ከንቲባ ስማቸው ማን ይባላል?	WR					WR					WR	WR	WR		
የደሴ ከተማ ፖሊስ መምሪያ የህብረተሰብ አቀፍ ወንጀል መከላከል ዋና የስራ ሂደት ባለቤት ማን ነው?	NA					√	√	NA			√	√			
ንብ ኢንሹራንስ ካሳ የከፈለው ለማን ነው?	WR					WR					WR				
የንብ ኢንሹራንስ ቦርድ ሰብሳቢ ማን ይባላሉ?	√					√					√				
ከ51 ነጥብ አምስት ቢሊዮን ብር በላይ ካፒታል ላስመዘገቡ ፕሮጀክቶች ፈቃድ የሰጠው ማን ነው?	WR	WR	WR			WR	WR	WR			WR	WR			
የኢትዮጵያ ኢንቨስትመንት ዔጀንሲ ፕሮሞሽንና የህዝብ ግንኙነት መምሪያ ዳይሬክተር ማን ይባላሉ?	WR	WR	WR			WR	WR	WR	WR		WR	WR			
ያለምርጫ ዴሞክራሲ የለም ያለው ማን ነው?	WR					√					WR				
የኢትዮጵያ ስኳር ዔጀንሲ ማን ይባላል?	WR					√					√				
በንግድና ኢንዱስትሪ ሚኒስቴር የማስታወቂያና የህዝብ ግንኙነት ጽ/ቤት ሀላፊ ማን ነው?	√					√		NA			√				
የ3ዔም ኢንጅነሪንግ ኮንስትራክሽን ፒ ዔል ሲ ዋና ስራ አስኪያጅ ማን ይባላል?	√					√					WR				
ፍሬሰላም የወተትና ወተት ተዋጽኦ ማህበር ሊቀመንበር ማን ይባላሉ?	WR					WR					√				
የሶማሊያ ፕሬዝዳንት ማን ይባላሉ?	√					√	√				√				
የኮሪያ አለማ ቀፍ የልማት ትብብር ዔጀንሲ ዋና ተወካይ ማን ይባላሉ?	WR					√					WR				
የህብረት ባንክ ፕሬዝዳንት ማን ይባላል?	WR					√					WR				
ፕሬዚዳንት ዶክተር አሸብር ለስንት ደቂቃ ንግግር እንዲያደርጉ ተጋበዙ?	√	WR				√	WR				√	WR			
የዶክተር አሸብርን የመልቀቂያ ንግግር ተከትሎ የዶክተሩ የውሳኔ እርምጃ ያልጠበቁት መሆኑን የገለጹት ከፊፋ የተወከሉት ሰዎች ስንት ናቸው?	√	WR				WR					√				
የአሜሪካ አለም አቀፍ የልማት ድርጅት ያቀረበው አስቸኳይ ጊዜ እርዳታ ምን ያህል ነው?	√					√					√	WR			
ደረጀ ኤቢሳ እድሜው ስንት ነው?	√	WR	WR			√					√				
ቶፕ ኮንስትራክሽን ለስንት ስራ አጥ ወጣቶች የስራ እድል መክፈት ቻለ?	√					√					√				
ቶፕ ኮንስትራክሽን ስንት አባላት አሉት?	√	WR				WR	WR				WR				
ባለፉት ሁለት አመታት የነበረው መምህር ተማሪ ጥምረት ስንት ነበር?	√					√					WR				
ባልፉት ዘጠኝ ወራት ስንት የአንደኛ ደረጃ መጻህፍት ታተሙ?	WR	√				WR					WR				

የኢትዮጵያ የአየር ትራፊክ ትቆጣጣሪዎች መሀበር ስንተኛ አመቱን ነው ያከበረው?	√					WR					WR				
ሰለ አፄ ገላውዴዎስ ሞት በግእዝ የተፃፈው በስንተኛው ክፍለ ዘመን ነው?	NA					√					√				
ዶክተር አሸብር ወልደ ጊዮርጊስ ስልጣናቸውን ሲለቁ ለምን ያህል ደቂቃ ንግግር አደረጉ?	√					√					√				
በፊፋ ፍኖተ ካርታ መሰረት ዶክተር አሸብር የቀጥሉ አይቀጠሉ የሚለው ሞሽን አንቀፅ ስንትን መሰረት ያደረገ ነው?	√	√				√	√				WR				
ግንቦት 8/2001 በተካሄደው የጠቅላላ ጉባዔ ከፊፋ የመጡት ተወካዮች ስንት ናቸው?	√					WR					√				
የአሜሪካ አለም አቀፍ የልማት ድርጅት ለጥያቄው ምን ያህል ግምት ያለው እርዳታ አቀረበ?	√					√					WR				
የአሜሪካ አለም አቀፍ የልማት ድርጅት ምን ያህል የምግብ እርዳታ ሰጠ?	√					√					WR				
ሲአር ቢሲ የሚገነባው መንገድ ርዝመቱ ምን ያህል ነው?	√					WR					WR				
ሲአር ቢሲ የሚገነባው መንገድ ስፋቱ ምን ያህል ነው?	WR					WR					WR				
ለብሔራዊ ፈተና ስንት የመፈተኛ ጣቢያዎች አሉ?	√					WR					√				
ለስንት የውጭና የሀገር ውስጥ ኢንቨስትመንት ፕሮጀክቶች ፈቃድ ሰጠ?	√					√					√				
በየአመቱ ስንት ቤቶች ለመገንባት እቅድ ወጥቶአል?	√					√					WR	√			
ስንት ፓርቲዎች ህጋዊ እውቅና አግኝተዋል?	√					√					√				
ስማቸው በምርጫ ቦርድ የሰፈረው ፓርቲዎች ስንት ናቸው		√				√					NA				
የቶፕ ኮንስትራክሽን ሀላፊነቱ የተወሰነ የህብረት ማህበር የአባላት ቁጥር ስንት ነው?	NA	WR				WR					WR				
የደቡብ ክልል ትምህርት ቢሮ ምን ያህል የ1ኛ ደረጃ ተማሪዎች መጽሀፍ አሳተመ?	WR					WR					WR				
የኢሳቅ ጎሳ አባላት አካባቢያቸውን ለስንት ሰአታት ይጠብቃሉ?	√					√					√				
የጋሻሞ ወረዳ ከሶማሌ ላንድ በስንት ኪሎ ሜትር ትርቃለች?	√					√					√				
የሾላ ምግብ ዝግጅት ባልትና አባላት ቁጥር ስንት ነው?	WR					WR					WR				
የሾላ ምግብ ዝግጅት ባልትና የደረቅ እንጀራ አቅርቦት በቀን ስንት ነው?	√					WR					√				
የሾላ ምግብ ዝግጅት ባልትና የአባላት የወር ገቢ ስንት ነው?	√					√					WR				
አንበሳ የከተማ አውቶቡስ ምን ያህል አውቶቡሶችን ጠገነ?	WR					WR					WR				
የአንበሳ አውቶቡስ አድሱን አሰራር የጀመረው በስንት መስመር ነው?	√					√					WR				
ከቁራ የሚነሳው አውቶቡስ ስንት ቁጥር ነው?	√	√				√					√	WR			
ከቁራ አውቶቡስ ተራ ድረስ ስንት ሳንቲም ያስከፍላል?	√					√					WR				
አንበሳ አውቶቡስ በአሁኑ ሰአት በምን ያህል አውቶቡሶች እየሰራ ነው?	√					WR	WR	WR			WR	WR	WR		

የዘንድሮ ብሄራዊ ፈተናዎች በአጠቃላይ ስንት የመፈተኛ ጣቢያዎች አሉት?	√					√					√				
በአድስ አበባ ውስጥ በአመት ከስንት በላይ ቤቶች ለመገንባት ታቅዷል?	WR					WR					√				
በአሁኑ ወቅት ያለው የአንድ ኪሎ ስኳር ዋጋ ስንት ነው?	√					WR					WR				
የኢትዮጵያ ስኳር ዔጀንሲ ያስቀመጠው የመነሻ እና የመድረሻ ዋጋ በየ ስንት ሳምንቱ ይቀያየራል?	√					WR					√				
በበጀት አመቱ የዘጠኝ ወራት ውስጥ ስንት ፕሮጀክቶች የኢንቨስትመንት ፈቃድ ወስደዋል?	√					WR	WR	WR			WR	WR	WR		
ፍሬሰላም የወተትና ወተት ተዋጽኦ ማህበር ስንት ላሞች ገዛ?	√					WR					WR	WR	WR		
ፍሬሰላም የወተትና ወተት ተዋጽኦ ማህበር አባላት ገቢ ስንት ነው?	WR					WR					WR				
የአለም ባንክ ስራ አስፈጻሚዎች ለኢትዮጵያ ምን ያህል ገንዘብ ለመስጠት አስበዋል?	√					√					WR				
ህብረት ባንክ በተያዘው የበጀት አመት ስንት ዶላር አገኘ?	WR					√					WR				
የብሮድ ባንድ ኢንተርኔት አገልግሎት የመመዝገቢያ ክፍያን በስንት አሻሻለ?	√					√					WR				
የጋሻሞ ወረዳ ከሶማሌ ላንድ በስንት ኪሎ ሜትር ትርቃለች?	√					√					√	√			
ሺሻ ሲያጨስ የተገኘ ግለሰብ ምን ያህል ብር ይቀጣል?	√					√					√				
የደሴ ከተማ ፖሊስ ስንት ቤት ፈትሿል?	√					WR					WR				
አትሌት መሰረት ደፋር የሴቶች የአምስት ሺህ ሜትር የቤት ውስጥ ውድድር በምን ያክል ሰዓት አሸነፈች?	√					√					√				
ቴዲ አፍሮ ለምን ያክል ጊዜ በስር ቤት ይቆያል?	√					WR					WR				

Appendix D: List of Main Java class files

No.	Class Name	Descriptions
1.	AnalyzeQuestion	Analyzes a question; Determine the question type, expected answer type, and question focuses
2.	AQAMain	The main class for the GUI. Accepts a question from the user and send the question to the remaining classes and finally return result to user.
3.	DatePattern	Extracts any date related answer particles from a document.
4.	FileIndexing, SenetcnceIndexing, ParagraphIndexing	Creates file, sentence, and paragraph based Lucene Indexes respectively.
5.	DocumentNormalizer	Normalizes the document for punctuation mar, charcter, and number variations.
6.	NameExtratcor	Gazetteer based name extraction
7.	NumberExtractor	Extracts numeric answer particles from a sentence.
8.	OneAnswerSelcetor	Selects one candidate answer from a document
9.	PatternBasedNameExtractor	Extarcts person name based on title patterns
10.	PlaceExtractor	Extarcts Place answer particles from a document
11.	QueryGenerator	Generates preoper IR query from a question
12.	Ranker	Ranks documents based on the answer particle in a document and distance computations
13.	TopAnswersSelector	Selects the best 5 answers from the ranked documents

Apendix E: Ethiopic Unicode representations (1200 – 137F)

	120	121	122	123	124	125	126	127	128	129	12A	12B
0	ሀ	ሐ	ሠ	ሰ	ቀ	ቐ	በ	ተ	ኀ	ነ	አ	ኰ
1	ሁ	ሑ	ሡ	ሱ	ቁ	ቑ	ቡ	ቱ	ኁ	ኑ	ኡ	
2	ሂ	ሒ	ሢ	ሲ	ቂ	ቒ	ቢ	ቲ	ኂ	ኒ	ኢ	኱
3	ሃ	ሓ	ሣ	ሳ	ቃ	ቓ	ባ	ታ	ኃ	ና	ኣ	ኲ
4	ሄ	ሔ	ሤ	ሴ	ቄ	ቔ	ቤ	ቴ	ኄ	ኔ	ኤ	ኳ
5	ህ	ሕ	ሥ	ስ	ቅ	ቕ	ብ	ት	ኅ	ን	እ	ኴ
6	ሆ	ሖ	ሦ	ሶ	ቆ	ቖ	ቦ	ቶ	ኆ	ኖ	ኦ	
7	ኋ	ሗ	ሧ	ሷ	ቇ		ቧ	ቷ	ኇ	ኗ	ኧ	
8	ለ	መ	ረ	ሸ	ቈ	ቘ	ቨ	ቸ	ኈ	ኘ	ከ	ኸ
9	ሉ	ሙ	ሩ	ሹ			ቩ	ቹ		ኙ	ኩ	ኹ
A	ሊ	ሚ	ሪ	ሺ	ቊ	ቚ	ቪ	ቺ	ኊ	ኚ	ኪ	ኺ
B	ላ	ማ	ራ	ሻ	ቋ	ቛ	ቫ	ቻ	ኋ	ኛ	ካ	ኻ
C	ሌ	ሜ	ሬ	ሼ	ቌ	ቜ	ቬ	ቼ	ኌ	ኜ	ኬ	ኼ
D	ል	ም	ር	ሽ	ቍ	ቝ	ቭ	ች	ኍ	ኝ	ክ	ኽ
E	ሎ	ሞ	ሮ	ሾ			ቮ	ቾ		ኞ	ኮ	ኾ
F	ሏ	ሟ	ሯ	ሿ			ቯ	ቿ		ኟ	ሐ	
	12c	12d	12e	12f	130	131	132	133	134	135	136	137
0	ዀ	ዐ	ዠ	ደ	ጀ	ጐ	ጠ	ጸ	ፀ	ፐ	※	፩
1		ዑ	ዡ	ዱ	ጁ		ጡ	ጹ	ፁ	ፑ	፡	፪
2	ዂ	ዒ	ዢ	ዲ	ጂ	ጒ	ጢ	ጺ	ፂ	ፒ	።	፫
3	ዃ	ዓ	ዣ	ዳ	ጃ	ጓ	ጣ	ጻ	ፃ	ፓ	፣	፬
4	ዄ	ዔ	ዤ	ዴ	ጄ	ጔ	ጤ	ጼ	ፄ	ፔ	፤	፭
5	ዅ	ዕ	ዥ	ድ	ጅ	ጕ	ጥ	ጽ	ፅ	ፕ	፥	፮
6		ዖ	ዦ	ዶ	ጆ		ጦ	ጾ	ፆ	ፖ	፦	፯
7			ዧ	ዷ	ጇ		ጧ	ጿ	ፇ	ፗ	፧	፰
8	ወ	ዘ	የ	ጀ	ገ	ኘ	ጨ	ፀ	ፈ	ፘ	፨	፱
9	ዉ	ዙ	ዩ	ጁ	ጉ	ኙ	ጩ	ፁ	ፉ	ፙ	፩	፲
A	ዊ	ዚ	ዪ	ጂ	ጊ	ኚ	ጪ	ፂ	ፊ	ፚ	፪	፳
B	ዋ	ዛ	ያ	ጃ	ጋ	ኛ	ጫ	ፃ	ፋ		፫	፴
C	ዌ	ዜ	ዬ	ጄ	ጌ	ኜ	ጬ	ፄ	ፌ		፬	፵
D	ው	ዝ	ይ	ድ	ግ	ኝ	ጭ	ጽ	ፍ		፭	
E	ዎ	ዞ	ዮ	ጆ	ጎ	ኞ	ጮ	ፆ	ፎ		፮	
F	ዏ	ዟ	ዯ	ጇ	ጏ	ኟ	ጯ	ፏ	ፏ		፯	

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.

Declared by:

Name: _____

Signature: _____

Date: _____

Confirmed by advisor:

Name: _____

Signature: _____

Date: _____

Place and date of submission: Addis Ababa, June 2009.