ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCES

Natural Language based Semantic Question Answering over Linked Data for Amharic Language

Gashaw Demlew Ayalew

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment for the Degree of Master of Science in Computer Science (Data and Web Engineering)

Addis Ababa, Ethiopia

Date: June 2019

ADDIS ABABA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

Gashaw Demlew Ayalew

Advisor: Fekade Getahun (PhD)

This is to certify that the thesis prepared by *Gashaw Demlew*, titled: *Natural Language based Semantic Question Answering over linked Data for Amharic Langauge* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science (Data and Web Engineering**)** complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Name	Signature	Date
1. Fekade Getahun (PhD), Advisor	_____	_____
2. _____	_____	_____
3. _____	_____	_____

DEDICATED TO:

Mother (Ayelu Tefera)

Acknowledgments

Most of all, I would like to thank God, who makes everything possible, for helping me pass all those hard times that I will never forget in my life.

I owe my deepest gratitude to my advisor Dr Fekade Getahun for his time, patience and undeniably helping comments all the way through this study. He really was an inspiration for me to proceed whenever I face difficulties and he is easily approachable. This thesis would not have been possible without his constructive comments on every aspect of the study.

My appreciation also goes to Teferi Kebebew (Msc) and staff members of faculity of Computing, Jimma Univesity, for spending their valuable time to respond to my questions and for providing the necessary data. Besides, I am very grateful to Jimma University Amharic teachers and students for spending their valuable time to respond to linguistic questionnaire forms.

I want to express my heartfelt thanks to Melkamu Beyene (PhD) for his unreserved assistance throughout this study. I would also like to express my appreciation to my family members and friends who have helped me in so many ways.

I really like to pass my sincere gratitude to my wonderful classmates for creating such an exciting batch. I will always cherish the time we have spent together. Finally, I want to thank all the people who have contributed in one way or another on this thesis work.

Abstract

As an enormous amount of structured data has been produced on the Web and available on online data portals in Amharic language, intuitive ways of accessing this data has become more and more important. Therefore, some question answering approaches have been proposed for other languages by researchers so far. However, as these approaches are language specific, they are not capable enough to capture grammar construction and statement formation of the Amharic language. On the other side, various researches have been proposed to retrieve for information from large repositories of Amharic text documents via using keyword-based search and semantic-based search. But they have lack of delivering direct information to the user; instead, they retrieve documents containing the needed information which user must scan to get information.

In this research, an effort has been made to design a new approach that allows the user to formulate a question in Amharic natural language using their own terminology to which they receive direct answers. Word embedding, Data indexing, Query template generation, Resource matching, and disambiguation, and Query ranking and execution are core components of the approach. Word Embedding component is responsible to construct vector representation of words based on the statistical distribution of words co-occurrence in an Amharic text corpus. Data indexing is intended to build indices for the purpose of speeding up the resource matching. Query template generation is responsible to interprets user query using the neural based semantic parser and generates the corresponding domain independent query template. Resource matching and disambiguation is intended to grounding domain independent query template to a given linked dataset through matching resources and disambiguating datasets to produce domains specific queries. This component produces several possible query templates which they are ranked and the top-ranked query is selected to retrieve answers via Query ranking and execution

The approach is evaluated using test data benchmark on Amharic linked dataset. The benchmark encloses 50 Amharic questions annotated with corresponding query templates and answers. It achieved average recall of 0.58, average precision of 0.43, and average f-measure of 0.50.

Keywords: Semantic querying, Question answering, Neural Word embedding, Neural semantic parser, Semantic matching, Resource disambiguation, Template generator.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Acronyms and Abbreviations

BGP:	Basic Graph Pattern
BLEU:	Bilingual Evaluation Understudy Score
BOA:	Bootstrapping Linked Data
CBOW:	Continuous Bag of Word
DeepQA:	Deep Question Answering
DUDES:	Underspecified Discourse Representation Structure
DSM:	Distributional Semantic Model
EAV:	Entity Attribute Value
ESA:	Explicit Semantic Analysis
FunQL:	Functional Query Language
IR:	Information Retrieval
IDF:	InverseDocument Frequency
KB:	Knowledge Base
KG:	knowledge Graph
LF:	Logic Form
LSA:	Latent Semantic Analysis
LSTM:	Long Short Term Memory
LTAG:	Lexicalized Tree Adjoining Grammar
MRR:	Mean Reciprocal Rank
NL:	Natural Language
NLP:	Natural Language Processing
NSPM:	Neural Semantic Parser Model
OWL:	Web Ontology Language
POS:	Part of Speech Tagger
PODS:	Partial and Ordered Dependency Structure
QA:	Question Answering
QALD:	Question Answering Over Linked Data
R2RML:	Relational to relational Mapping Language
RDB2RDF:	Relational Database to RDF
RDF:	Resource Description Framework
RIF:	Rules Interchange Format
RNN:	Recurrent Neural Network
RSS:	Relation Similarity Service

SCoDD:	Size, Complexity, Dynamicity and Decentralization
SG:	Skip Gram
Seq2Seq:	Sequence to Sequence
SPARQL:	SPARQL Protocol and RDF Query Language
SOV:	Subject Object Verb
SPSS:	Statistical Package for Social Science
SVO:	Subject Verb Object
TF:	Term Frequency
TF /IDF:	Term Frequency/ Inverse Document Frequency
TP:	Triple Pattern
Word2Vec:	Word to vector
WIC:	Walta Information Center
WVM:	Word Vector Model

Chapter 1 – Introduction

1.1 Overview

Since digital technology has emerged, a massive amount of data has been produced in a structured format. Large data is available from online transactions, social networking sites, statistical data, online data portals, and etc. In addition, data which had been held in paper or hard copy for a long time has been converted to soft copy to share them with the public. For instance, the Ethiopian government as part of the eOffice project to decide to change part of the paper-based reports to be converted into its equivalent electronic version. Especially, the Ethiopian government developed an open data Web portal that allows organization to publish their data [1]. These large collections of data contain a set of facts in multiple domains.

On the other hand, the DBpedia community project extracts very large, structured, semantically rich knowledge from Wikipedia and makes it freely available on the Web using Semantic Web and Linked Data technologies. The project extracts knowledge from different language editions of Wikipedia and releases all DBpedia knowledge bases for use [2]. One of the releases, Amharic DBpedia, is extracted from the Amharic version of Wikipedia, provides a wealth of human knowledge across different domains [3]. Linking the above-mentioned data sources to Amharic DBpedia results in a large collection linked dataset that allow user to acquire valuable information across open domains. But ways of accessing information from these large collections of data has become a critical issue.

A number of formalisms have been proposed for representing data and metadata on the Semantic Web. In particular, RDF, Topic Maps and OWL allow one to describe relationships between data items, such as concept hierarchies and relations between the concepts. A key requirement for the Semantic Web is integrated access to data represented in any of these formalisms, as well the ability to also access data in the formalisms of the ‘‘standard Web,’’ such as HTML and XML. A wide range of query languages for the Semantic Web exist, ranging from (i) pure ‘‘selection languages’’ with only limited expressivity, to fully-fledged reasoning languages, and (ii) from query languages restricted to a certain data representation format, such as XML or RDF, to general purpose languages that support multiple data representation formats and allow simultaneous querying of data on both the standard and Semantic Web [4].

Semantic queries allow for querying and analytics of associative and contextual nature. They enable the retrieval of both explicitly and implicitly derived information based on syntactic, semantic and structural information contained in data. They can deliver precise results (possibly the distinctive selection of one single piece of information) or answer more fuzzy and wide open questions through pattern matching and digital reasoning. Semantic queries work on named graphs, linked-data or triples. This enables the query to process the actual relationships between information and infer the answers from the network of data. This is in contrast to semantic search, which uses semantics (the science of meaning) in unstructured text to produce a better search result. Semantic queries are used in a triple store, graph databases, semantic wikis, and natural language and artificial intelligence systems.

A Natural Language query consists only of normal terms in the user's language, without any special syntax or format. It allows a user to enter terms in any form, including a statement, a question, or a simple list of keywords. The query can be simple, Complex, compound. A simple query consists of a single and unconstrained query-desire (explicit or implicit) and a single, unconstrained, and explicit query-input. For example, in "የኢትዮጵያ ዋና ከተማ ማነው?" the query desire (i.e. ዋና ከተማ) is explicit, single, and not constrained by any clausal phrase. The query input (i.e. ኢትዮጵያ) is also explicit, single, and unconstrained. A complex query consists of a single query-desire (explicit or implicit, constrained or unconstrained) and multiple, explicit query-inputs (constrained or unconstrained). For example, the query "የኢትዮጵያ ዋና ከተማ በዓፄ ምኒልክ ዘመነ መንግስት ማነው" is a complex query where the explicit, unconstrained query desire is single (ከተማ) while the query input is multiple (ኢትዮጵያ, ዓፄ ምኒልክ ዘመነ መንግስት). A compound query consists of a conjunction/disjunction operator connective between one or more simple or complex queries. This thesis will focus on all types' queries.

While typing queries in formal query languages provide the greatest level of expressivity and control for the user, it is also the least user-friendly access interface. Query languages have complex syntax; require a good understanding of the representation schema, including knowledge of details like namespaces, class and property names. So, exploring the Web of data by structured query languages is tedious and error-prone.

Various researches [5, 6, 7] have been undertaken in the area of searching and retrieving information from large repositories of Amharic text documents. To do these activities they used different information retrieval (IR) techniques; keyword-based search and

semantic-based search over unstructured text document. But the existing searching system over Amharic documents have lack of delivering readymade (or direct) information to the user, instead, they retrieve documents containing the needed information, which the user must then scan to search for pertinent information. The purpose of this thesis is to come up with a new searching approach dedicated to Amharic Web of data that could show improvement over the existing searching engines by semantically processing user query and retrieving direct answer over the open domain.

Generally, this thesis will provide an approach to retrieve information from a large collection of linked data, such as the Web of data. The approach allows users to formulate a query in Natural Language (NL) using their own terminology to which they receive a concise answer generated by querying a knowledge base. Users are thus free from two major access requirements of the Semantic Web: (1) the mastery of a formal query language and (2) knowledge about the specific vocabularies of the knowledge base they want to query. Many types of research related to this thesis have already been developed for the English language [10, 11, 12]. However, these researches do not address the basic characteristics of the Amharic language. This thesis is about Amharic natural language query answering over Amharic linked data.

The Amharic language has its specific way of grammatical construction, character (Fidel) representation and statement formation [15, 16] where system depends on query processing. So, the query construction and answering techniques in the Amharic language are different from English and other languages. The research work will attempt to identify the basic language specific issues in query processing.

1.2 Motivation

Recently, the Ethiopian government developed an open data portal and allowed government offices to participate in publishing structured data as open data on the Web. As many organizations participating on the portal, the size of these data has increased and handles a set of facts in multiple domains [1]. These data sets together with the Amharic DBpedia and other data sources lead to the user can acquire any kind of information in the open domain. So there is a need for a new way to integrate and access these large collections of structured data.

The traditional Information retrieval techniques were considered insufficient in retrieving precise information to the user [5, 6, 9]. 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 and secondly, they want their results to be real answers - not the list of relevant document links. In-line with these two prominent researches been conducted in the area of Amharic question answering. The first one is "TETEYEQ: Amharic Question Answering for Factoid Questions" [9]. It uses Keyword-based information retrieval (IR) technique to process large amounts of unstructured text and as such it retrieves the documents and paragraphs in which the answer might appear. However, the approach can't capture the implicit relation among terms or the semantics of words in the document. Thus, the system will respond no result to the user (low recall). Besides, it retrieves an irrelevant document to the user (low precision). The second is "Amharic Semantic Search Engine" [5]. It handles the semantics of words but is limited to searching on a single domain (i.e. searching only one kind of information). But there are user's queries which are much complex and need integration of information from different related data sources.

The high heterogeneity of linked-data makes it difficult for humans to search and discover relevant information. As linked data is in RDF format, the standard approach would be to run structured queries in triple-pattern-based languages like SPARQL. However, such language is used only by expert programmers and knowledge Engineers who have a deep understanding of the structure of the knowledge base whereas the use of natural language is relatively convenient from the user sides. For instance, a user poses the query *"ከአጼ ኃይለ ሥላሴ ባለቤት የሚወለዱት ልጆች".* The answer could be found by querying several linked data sources, about Haile Selassie and Menen Asfaw. A possible SPARQL triples formulation: *<#resource/ኃይለሥላሴ #property/ባለቤት ?x> . < #resource/ኃይለሥላሴ #property/ልጆች ?y> . < ?x #property/ልጆች ?y>*. This query involves multiple joins, would yield good results, but it is difficult for the user to come up with the precise choices for relations, classes, and entities. This would require familiarity with the contents of the knowledge base, where the average user is not expected to have.

The grammar construction of the user's query in the Amharic language is different from English and other languages. Simple case, in English, questions will be developed, for example, using "Wh" words such as "who is the Prime Minister of Ethiopia?" and so on. But the same question has a different structure in Amharic due to the difference in character, word formation, grammatical arrangement, and type of question particles

(terms used to ask questions) used, as (የኢትዮጵያ ጠቅላይ ሚኒስተር ማን ይባላሉ ye-Ethiopia Teqlay minister man yibalal). In addition, a user may use different question particles for one question (e.g. ይባላል፣ይባላሉ…).

The Amharic Language has a wide range of complexity such as lexical ambiguities (a single word is associated with multiple senses or meanings) and complex syntactical structures (i.e. morphologically complex). Understanding of Amharic natural language query has the following challenges; the first is the accurate identification of the query desire of a user-query. Another challenge is that a query can be paraphrased into multiple forms, thereby triggering the potential of lexico-syntactic mismatches. Also, there is no unique way to create schemata in RDF, i.e. the same fact can be written in different triple forms and could also be expressed using multiple triples.

Given these motivations, the aim of this thesis is to design a new natural language based question answering approach for Amharic, users have thus freed the mastery of a formal query language and knowledge about the specific vocabularies of the knowledge base they want to query.

1.3 Statement of the Problem

Large data is available from online transactions, social networking sites, statistical data, online data portals, and etc. Especially, the Ethiopian government developed an open data Web portal that allows the organization to publish their data. These large collections of data contain a set of facts in multiple domains; it leads to the user can acquiring any kind of information in the open domain. The problem is how this information can be integrated and accessed to fill the user's information need. As a solution, various researches [5, 6, 7] have been undertaken in the area of searching and retrieving information from large repositories of Amharic text documents.

To do these activities they used different information retrieval (IR) techniques; the most common techniques are keyword-based search and semantic-based search. Keyword-based search can't capture the implicit relation among terms or the semantics of words in the document. Thus, it will respond no result to the user (low recall). Besides, it retrieves an irrelevant document to the user (low precision). semantic-based search, which go beyond keyword based search is based on concept search, focuses on the text and limited to searching on a single type of data in a single domain (i.e. searching on only one kind of information), and require less (or no) resolving of ambiguity as system searching on

open domain. But there are user's questions which are much complex, need integration of information from different related data sources.

Because of these searching approaches, the existing searching system over Amharic documents have lack of delivering readymade (or direct) information to the user, instead, they retrieve documents containing the needed information, which the user must then scan to search for pertinent information. So we need another system, technology and approach to solve the above-mentioned problems to satisfy user's information need.

The intention of this research is to come up with a new semantic question answering approach to find information needs of user from Amharic linked data, where it is based on the technologies involving on structured data; *semantic Web*, and *linked data*. It is expected to eliminate the problems of the existing Amharic question answering by reducing the effort involved in formulating effective queries for search engines and making results to be real answers - not the list of document links. The research work lies on designing and developing a new natural language based semantic question answering approach over Amharic linked data. It allows users to express arbitrarily complex information needs in natural language using their own language.

Thus, the key research questions associated with this thesis work are:

- ✓ What kind of approach is appropriate to find information from a large collection of structured data? (**semantic questioin answering/schema agnostic quering**)
- ✓ How can we handle and represent the internal structure and meaning of a natural language query? (**Deep semantic parsing**)
- ✓ What model is important to capture data in the knowledge base? (**RDF data model, linked data)**
- ✓ What kind of approach is appropriate to handle lexical/semantic approximation problems (**Distributional semantic model**)
- ✓ How can we integrate information from different dataset resources to fill user information need? (**Resource Disambiguation**)

1.4 Objectives

General Objective

The overall objective of this research is to design and implement a natural language based semantic question answering system over linked data for Amharic langauage.

Specific Objectives

To accomplish the abovementioned general objective, the following specific objectives are devised.

- ✓ Assessing approaches that are used by existing search engines and their limitations.
- ✓ Analyzing grammatical and syntactical structures of Amharic natural language.
- ✓ Building up an indexing structure by integrating word vector space with TF/IDF of RDF documents.
- ✓ Collecting structured data and prepare them into forms which linked data based.
- ✓ Collecting Amharic question and annotating them via their corresponding logic form
- ✓ Designing a deep neural network based semantic parser.
- ✓ Designing semantic based question .
- ✓ Providing an approach that integrates information from different sources.
- ✓ Implementing the proposed solution
- ✓ Evaluating the proposed solution using evaluation metrics.

1.5 Methods

In order to accomplish the general and specific objectives of this study, different methodologies will be applied.

Literature Review

Extensive literature reviews will be conducted to understand the various components of the research. Specifically, literature in the area of interpreting complex natural language query (methods), Amharic query grammatical and syntactical structures, lexical ambiguities resolution techniques, and existing searching approaches and search engine researches for Amharic and non-Amharic languages will be reviewed. Reading and criticizing all these literature and related works helps to gain the required knowledge on a certain subject and also assists in identifying the right methods and tools for implementing the different components of the system.

Data Sources for the Experiment

To test the proposed system, we will use three different sets of data to undertake experiments. The first is a collection of structure data which will be collected from

different sources such as the government's open data portal, online transactions, statistics data, and online data portals. The collected data will be mapped to RDF format and then linked to Amharic DBpedia, Amharic linked data. The dataset contains 30,000 RDF triples. The second set is a collection of Amharic texts which will gather from different sources including Amharic news from WIC information center, Amharic Wikipedia dump and generic version of the bible. It is used to build distributional semantic models. Total of 35,877 Amharic text files will be collected. The third set of data is a collection of Amharic questions and their corresponding annotations. It is used to build and test deep neural semantic parser and to test the overall system performance.

Experimentation

In order to accomplish the objectives of the research, different methods and tools will be engaged in the process. Data from different sources stored in the relational data model, we will use an open source RDF mapping tool like RDB2RDF, R2RML, to translate them into RDF data model. Java and Python programming languages will be used as a major development tool of system prototype, and we will integrate the Jena framework to manipulate the knowledge base. Table database integrated inside Jena will be used as a backend to store linked data. The SPARQL (RDF query language) is used as a query language. Lucene API (version 3.7) will use for data indexing.

Testing and Evaluation

To evaluate the performance of the proposed solution, the system will be tested using Amharic linked data collected from different sources, and all query types (simple, complex, and compound). The relevance of the resulted answer will be evaluated using statistical relevance method recall, precision, and F-measure techniques.

1.6 Scope and Limitations

Scope

This research has been conducted to provide user-friendly access interface that allow users to formulate a question in their own language and to deliver direct answer through semantically quering large collection of linked data for Amharic language.

Limitation

- ✓ The searching approach does not work on textual document data to search answer for user query.

- ✓ Processing of query needs an understanding of natural language. It requires implementations of other NLP tools such as chunker, NER, entity linker and so on. Some of the tools have been built as prototypes; they are not fully implemented. As a result, the interpretation of the query will not be accurate.
- ✓ The data quality including data incompleteness is not handled during searching (i.e. data inference). It will affect the performance.
- ✓ The templates are generated manually. So, the system will have limited coverage to answer question in open scenario.

1.7 Application of Results

The system can be applied in finding information for Amharic natural language queries from large collections of data, linked data. If it is transformed into a full-fledged system, it can be applicable as a search engine on the Government Open data web Portal to access information on different domains. In addition, the study can open a way for further researches in the area of Open Data and Linked Data to publish and interlinking structured data on the Web for the Amharic language, and hybrid Question Answering that requires the integration of information both from Structured and from textual sources.

1.8 Thesis Organization

The rest of the Chapters in this report are organized as follows. Chapter two deals with the literature reviewed so far. concepts of contemporary data management environments, different data models and criteria for the selection of reference data model, traditional search engine and its limitations, concept, and approach of semantic searching and its limitations, approaches of question answering system over the linked data, and the Amharic Language Writing System. Chapter three discusses the existing search engine/ question answering methods for Amharic and English Languages. Chapter four describes the Design and Implementation of the proposed Amharic question answering over linked data system. The Experiment and Evaluation of the system are discussed in Chapter five. Finally, conclusions are drawn from the thesis result, the contributions of this research work and recommendations on possible future works related to this research are given in Chapter six.

Chapter 2 - Literature Review

2.1 Introduction

In this Chapter, extensive reviews in the general concepts of data management environments, different data models and criteria for the selection of reference data model, traditional search engine and its limitations, concept and approach of semantic searching and its limitations, concept, challenges, and approaches of question answering (QA) system over Web of data, and finally details about Amharic language includes the alphabets, the punctuations, the numbers, and the Morphological complexity, the syntactic and the grammar structure of the language are presented. The extensive literature has been reviewed to understand the problem associated with the realm of this thesis and also to identify the appropriate solution.

2.2 Data Management Environments

The database (structured data) landscape is changing rapidly, influenced by the need to cope with databases with increasing schema size, complexity, dynamicity, and decentralization (SCoDD) conditions. The emergence of new data sources such as open datasets on the Web, sensor networks, data from mobile applications, social network data, together with the natural growth of datasets inside organizations [21], brings the demand for data management strategies which can operate under the properties dictated by this new data environment.

The challenges, new approaches, and trends for coping with this new data landscape are currently aggregated under the Big Data umbrella. According to [25], Big Data is the term for a collection of datasets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications. Propelled by the growth of the Web and on the number of available computational devices, data management requirements are shifting towards the need to cope with decentralized data generation [23, 24]: data which is intrinsically heterogeneous, with different structured levels and generated under different meaning contexts.

2.3 Data Models

The multiplicity of data model categories represents a natural specialization to cope with different data management requirements. According to Beat Sprenger [26], analysis,

formulation and evaluation of a query approach is grounded on a reference data model, where a selection of the reference data model obeys the following criteria:

- Ability to represent large, complex & heterogeneous conceptual models: Associated with the representation trends in data management
- Ability to represent and map other data models: Maximizing the generality and transportability of the results.
- Adoption in test collections suitable for schema-agnostic queries: Maximizing the comparability of the query approach.

The followings are a list of data models and in each sub-section; we observe how they are analyzed according to the criteria defined above.

2.3.1 Relational Databases

The relational model for database management is a data model based on first-order predicate logic, formulated and proposed by Codd [23]. In the relational data model data is represented as tuples and grouped into relations (i.e. table, a visual representation of a relation). The motivation behind the relational model is to provide a declarative method for specifying data and queries. The relational model emerged to define a model for describing data in terms of its natural structure, without superimposing any additional data management structure. The consistency of a relational database is enforced by the Database Management System (DBMS) for all applications [23], not by rules built into the applications that use it.

Despite its major adoption, relational databases have limitations for coping with the list of data model requirements**.** These limitations are described below**:**

- Ability to represent large, complex & heterogeneous conceptual models:
  - ✓ Limitation in the representation of sparse data. Relational models target the representation of compact (non-sparse) data. A relation is defined by a rigid set of attributes where the state of each attribute in a relationship needs to be defined by a value assignment, for each tuple.
  - ✓ Schema rigidity. Relational models are based on the concept of a prescriptive schema, which enforces the consistency of the data under a specific conceptual model. Relational models constrain the evolution of the schema.

- ✓ Complex data integration: The integration of relational databases under different schemas depends on a redefinition of the schema.

- Ability to represent and map other data models: Mapping is limited by the maximum number of attributes for a table.
- Adoption in test collections suitable for schema-agnostic queries; it is a query approach over structured dataset which allow users satisfying complex information needs without the understanding of the representation (schema) of the dataset. Its limitation is that the test collection for natural language over databases has less schema size, is not sufficient to adopt schema-agnostic queries in search approaches.

2.3.2 Semantic Web Databases/Linked Data

New data models emerged to facilitate the deployment of decentralized, large-schema, sparse and schema-less data environments. More prominently the effort associated with the creation of a Semantic Web / Linked Data Web has motivated and catalyzed the development of standardized data models which can support the schema size, complexity, dynamicity, and decentralization (SCoDD) conditions. The vision of the Semantic Web is grounded on standards-based data representation, consisting of layers of different knowledge representation concerns. We review them as follows

RDF: is a general-purpose language for representing information in the Web. It is particularly intended for representing information about Web resources. It is a framework for representing information about resources in a graph form. Information is represented by subject-predicate-object expressions. These expressions are known as triples in RDF terminology. The subject denotes the resource, and the predicate denotes traits or aspects of the resource and expresses a relationship between the subject and the object.

Several common serialization formats are in use for RDF, including [34]; Turtle: a compact, human-friendly format, N-Triples: a very simple, easy-to-parse, line-based format that is not as compact as Turtle, N-Quads: a superset of N-Triples, for serializing multiple RDF graphs, JSON-LD: a JSON-based serialization, N3 or Notation 3, a non-standard serialization that is very similar to Turtle, but has some additional features, such as the ability to define inference rules, and RDF/XML: an XML-based syntax that was the first standard format for serializing RDF.

Web ontology language (OWL): is a language for defining and instantiating Web ontologies [26]. OWL ontology may include descriptions of classes, properties, and their instances. Syntactically OWL ontology is a valid RDF document and as such also a well-formed XML document.

Ontology: is a specification of a conceptualization. Conceptualization means structured interpretation of a part of the world in which people used to think and communicate in the world. Ontology describes knowledge about the domain in terms of concepts or vocabularies within the domain and relationships between them. Depending on the scope of the ontology, ontology may be classified as generic ontology, domain ontology, task ontology and application ontology [25]:

- Generic ontology - describing general concepts that have a wide scope and are applicable for several domains. Also called, upper, top-level ontologies.
- Domain ontology - describing for a specific domain, such as the medical domain, engineering domain, or narrower domains, such as personal computers domain
- Task ontology- are ontologies suitable for a specific task, such as assembling parts together
- Application ontology - ontology developed for a specific application, such as assembling personal computers.

Simple Protocol and RDF Query Language (SPARQL): is a query language and data access protocol for the semantic Web [35]. SPARQL is defined in terms of the W3C's RDF data model and will work for any data source that can be mapped into RDF. A SPARQL query is composed of:

- The SELECT clause identifies the variables to appear in the query results and expression that indicates how to construct the answer to the query.
- The WHERE clause provides the basic graph pattern to match against the data graph

Rules Interchange Format (RIF): is a rule interchange format. Rules are statements that can be used to infer (discover) new knowledge. A typical kind of ontology for the Web has taxonomy and a set of inference rules. The taxonomy defines classes of objects and relations among them. The inference rules allow an application to make decisions based on the classes supplied without needing to actually understand any of the information provided [19].

The problems associated with ensuring logical consistency and reasoning performance at Web scale strongly limited the adoption and growth of the Semantic Web. In order to increase its adoption, Berners-Lee [28] proposed a simplification over the previous Semantic Web representation scheme, by concentrating on the data model representation layers (i.e. lower layers of the Semantic Web stack), focusing on the mapping and publication of existing datasets available in data silos. This simplification defined the vision of a Linked Data Web, targeting the creation of a Web of structured data based on a standardized data model (RDF(S)). Since the conception of the Linked Data Web vision, datasets covering different domains have been exposed as Linked Data.

Differently, from the relational model, which depends on schema-level data integration, the linked data Web is based on an entity-centric data integration model, where URIs representing objects or concepts can be reused or reconciled among different datasets. The entity-centric data integration facilitates the co-existence of different perspectives and points of views of entities and a decentralized evolution of the data.

The publication of Linked Data is summarized under the Linked Data principles [28]:

1. Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF(S), SPARQL)
4. Include links to other URIs, so that they can discover more things.

Linked Data is adopted in the evaluation of contemporary data environment challenges such as question answering, entity search, entity consolidation, among others. Here we examine it according to the criteria defined above.

- Ability to represent large, complex & heterogeneous conceptual models ([28]); support for the representation of sparse data: RDF(S) is based on a graph data model, which supports a sparse data model, schema flexibility: RDF(S) datasets are schema-less and can be evolved in a decentralized manner, entity-centric integration: Support for decentralized entity-centric data integration.
- Ability to represent and map other data models: Data in a relational or in a CSV format can be systematically mapped to RDF.

- Adoption in test collections suitable for schema-agnostic queries: Various test collections for Linked Data are available, under SCoDD conditions based on datasets with 1,000s or 1,000,000s of attributes.

2.3.3 Entity-Attribute-Value

The Entity-Attribute-Value model (EAV) is a data model to describe entities where the number of attributes (properties, parameters) that can be used to describe them is potentially vast, but the number that will actually apply to a given entity is relatively modest" [30]. From a practical perspective, the EAV model is associated with open/dynamic schema databases. EAV can be seen as the more abstract data model behind RDF(S) and Linked Data. An EAV data model is composed of three core elements [30, 31]:

- Entity: the element being described. In RDF(S) the entity maps to an instance.
- Attribute: the attribute definition. In RDF(S) it an attribute maps to a property or a class.
- Value: The value assigned to an attribute. In RDF(S) it maps to an object which can be an instance or value.

The generality of RDF(S) in the representation of data under the SCoDD characteristics, the ability of RDF(S) to map existing data models, including relational databases, CSV files, data log, key-value pairs, the availability of real-world datasets, the possibility of describing complex conceptual models using RDF Schema, and its support as a data model for a logical model, motivated the use of RDF(S) as the core data model and Linked data as data integration model, and schema-agnostic querying as query approach for grounding this work.

Next, we review and explain different approaches of searching/retrieving of information, their challenges, strengths, and limitations.

2.4 Traditional Search Engine

A search engine is a software system designed to search for information on the World Wide Web. It helps users to find relevant information from the Web, where a user submitted query as a sequence of terms that describe the content, he/she is interested, and in return, the search engine returns a list of potentially relevant Web documents. The evaluation metrics for search engine, generally IR, are performed by using a document

collection, a set of topics describing a user's information needs and a set of relevance judgments, indicating, for each topic, which documents are relevant for each topic.

Generally, search engines perform three basic tasks:

- They crawl the Web to collect Web documents based on important words.
- They keep an index of the crawled resources they find.
- They allow users to look for words or combinations of words found in that index.

And consist of three basic components the crawler, the indexer and the query manager [32]:

Crawler: is a program that traverses the Web to collect Web documents and stores them in a database. It gathers resources (HTML pages, multimedia contents, etc.) from the Web for later analysis by the indexer component.

Indexer: is the action of taking the resource gathered by the crawler and building a data structure to be used for searching. An indexer extracts information of document and put in the index file. The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query.

Query engine: is the component that interacts with a user. It accepts the user query, consults the indexer component, makes some ranking, and finally presents the results to the user. The query engine may also perform query expansion in consolidation with a dictionary that contains synonym of words.

Limitations of Traditional Search Engine

The current Web is the largest globally distributed database that lacks semantic structure and hence it makes difficult for the machine to understand the information provided by the user. The following are some of the key problems of the current Web and searching on the Web.

- The Web content lacks a proper structure regarding the representation of information
- The ambiguity of information resulting in poor interconnection of information.
- The incapability of machines to understand the provided information due to the lack of a universal format.

Another effort is introduced, semantic searching, to overcome the above limitation of the traditional search engine.

2.5 Semantic Searching

It attempts to augment and improve traditional search results by using data from the Semantic Web. The semantic search extends the scope of traditional information retrieval paradigm from document retrieval to entity and knowledge retrieval. Its goal is delivering information by taking into account the context of user search as well as the underlying meaning of the documents to be searched rather than a list of loosely related keyword results. The following are attributes of semantic search (i.e. qualities that make it distinct from non-semantic search) [28]: handing morphological variations, handling synonyms with correct sense, handling generalization, handling concept matching, handling knowledge matching, handling natural language queries and questions, ability to point uninterrupted paragraph and the most relevant sentence, ability to enter queries freely no special format like quotes or Boolean operators, ability to operate without relying on statistical user behavior, and other artificial means, and ability to detect its own performance

The following are some of the key problems of the current semantic search engines

- Knowledge overhead which is requiring users to be equipped with extensive knowledge of the back-end ontologies and knowledge bases (e.g. form-based search engines) or specific SQL like querying languages (e.g. RDF-based query language fronted search engines) in order to be able to formulate queries or to understand the search result.
- Lack of support for answering complex queries presented by current semantic-based keyword search engines.
- They are useful for searching on a single type of data (i.e. text) in a single domain. Therefore, they are incapable of answering complex questions involving the integration of information from collection of resources (i.e. multiple types of data from multiple sources).
- They can’t currently provide a level of query interpretation that could point directly to the final answer.

Semantic Web is introduced to overcome the above limitations of semantic search engines by providing a framework for storing, managing, sharing of data, and querying

across many types of related information. It allows answering complex questions by integrating information from multiple sources in multiple domains, unlike semantic search, allows searching in a single domain. As our goal is filling user's information need in the open domain, thus we are going to use the semantic Web framework for grounding our approach. In the coming section, we review the approaches and the key requirements necessary for querying/ searching semantic Web data.

2.6 Approaches for Querying Semantic Web/Linked Data Datasets

Different Authors [24, 49] describe the key requirements necessary for query processing on the Semantic Web.

- High usability & Low query construction time: Support for a simple and intuitive interface for experts and casual users.
- High expressivity: Queries referencing structural elements and constraints in the dataset (relationships, paths) should be supported, as well as operations over the data (e.g. aggregations, conditions).
- Accurate & comprehensive semantic matching: Ability to provide a principled semantic matching addressing all the dimensions of the semantic heterogeneity problem (abstraction, conceptual, compositional, functional). Semantic matching with high precision and recall.
- Low setup & maintainability effort: Easily transportable across datasets without significant manual adaptation effort. The query mechanism should be able to work under an open domain and across multiple domains.
- Interactive search & Low query-execution time: Minimization of user interaction/ feedback effort in the query process. Users should get answers with interactive response times for most of the queries.
- High scalability: The query approach should scale to large datasets both in query execution and indexing construction time. The query approach should scale to a large number of datasets.
- Semantic tractability mechanisms: the query approach should be able to answer queries not supported by explicit dataset statements (i.e. should support data inference).

- Ability to query distributed heterogeneous data: the ability to find the right sources that can answer user query and integrating partial answers found from different sources.

Beat Sprenger [26] categorizes approaches for querying/ searching semantic Web data into four:

a) Entity Search: Vector Space Models for Semantic Web/Linked Data Datasets

Entity search approaches utilizing techniques used in information retrieval for searching entities (instances and classes) over Semantic Web/Linked Data. The use of vector space models and the associated data structures (inverted indexes) is used to support query execution and indexing temporal performance and scalability. But this approach cannot afford full expressive natural language queries (not provide schema-agnostic queries with higher expressivity).

b) Approximate Queries for Semantic Web/Linked Data Datasets

Existing approaches under the approximate queries category concentrate on the employment of the relaxation of structural query constraints and on exploiting the semantic information on the dataset to support a taxonomic based semantic relaxation. In this approach, Users need to be aware of a partial structure of the vocabularies (low usability) and expressive queries are supported in the context of SPARQL (i.e. partial expressivity).

c) Visual Query Interfaces for Semantic Web/Linked Data Datasets

It concentrated on allowing users to build structured queries by exploring graphical user interface elements. In this case, users are not abstracted from the schema, but the visual interface allows users to do a visual exploration in a constrained subset of the data. In a schema-agnostic scenario, where users query open domain, large-schema datasets, this approach is likely to present scalability problems: the space of query guides can potentially become too large for users selecting query sub-patterns and for the computation of query suggestions.

d) Natural Language Interfaces for Semantic Web/Linked Data Datasets

NLI focuses on approaches which use open or controlled natural language queries for querying databases. It concentrated on approaches which provide higher levels of schema-agnosticism, better exploiting semantic techniques to address schema-

agnosticism (high usability and high expressivity). Although some approaches focus on the question-answering (QA) problem, in which precise answers are expected as the output. Others focus on a best-effort scenario; merge natural language queries' expressivity and usability with IR models' scalability that returns an approximate ranked list of results.

In our work, we focus on the first scenario (question answering) that allows user inquires information's need using thier own terminology (high usability) and retrieve top ranked answers by querying datasets as well as to operate over the data (aggregate results, express conditional statements, and so on). Accordingly, the next section assesses dimensions, existing approaches, and the challenges of question answering over semantic Web of data.

2.7 Question Answering (QA) System

QA is a specialized form of information retrieval, retrieves exact answers to questions posed in natural language by the user. It allows users to express arbitrarily complex information needs in an intuitive fashion and in their own language. In contrast to traditional search engines, question answering systems allow users to pose a (possibly complex) question, instead of merely a list of keywords, and return precise answers, instead of documents in which the answer can be potentially found [37].

Originally, question answering had a strong focus on textual data sources to find answers, relying mostly on information retrieval techniques. In the early 70's, question answering then started to incorporate structured data, developing natural language interfaces to databases [31]. Nowadays, with the growing amount of knowledge in the linked open data cloud, interest in question answering over structured data is quickly regaining interest.

2.7.1 Dimensions of Question Answering

Categorizing the different dimensions of question answering is fundamental for understanding the different phenomena and challenges involved in a question answering task and helps in the definition of the scope of a question answering system and to set up a proper evaluation. Beat Sprenger [26] categorizes the following dimensions involved in question answering:

- *Question and answer type***:** Classifying questions with respect to the answer form results in a question taxonomy roughly along the following lines: Factoid questions (including predictive questions, list questions, and yes/no questions), Definition questions, Evaluative or comparative questions, Association questions, Explanation/Justification questions, Process questions, and Opinion question.
- *Data sources:* Question answering systems differ with respect to the data source(s) they are able to process to derive an answer. On the one hand, they usually consume a specific type of data: structured data, semi-structured data, and unstructured data. On the other hand, question answering systems differ with respect to the number of data sources they consider: a single dataset, an enumerated list of multiple (i.e. distributed datasets), and all datasets available on a large scale (i.e. considering all datasets of a certain kind (e.g. structured or text) available on the Web or in the linked data cloud)

Furthermore, a question answering system can be either domain-specific, addressing very specific knowledge in a particular domain, e.g. the biomedical or financial domain, or open-domain, addressing general knowledge, especially encyclopedic knowledge, common-sense knowledge, news or trivia.

Typically, the kind of information represented in the linked data is important for answering factoid and definition question categories. Thus, our focus is answering such question types and addressing open domain to fill information need of user's.

2.7.2 Challenges in Question Answering over Web of Data

The followings are challenges to answer questions from the Web of data:

a) Understanding Questions

In the case of QA over structured data, the question must be translated into a logical representation that conveys its meaning in terms of entities, relations, types as well as logical operators. The task of translating from natural language to a logical form (i.e. semantic parsing) is characterized by the mismatch between natural language (NL) and knowledge base (KB). The semantic parsing problem can be divided into two parts:

- Determining KB constituents mentioned in the NL expression and
- Determining how these constituents should be arranged in a logical structure.

The first problem involves mapping natural language expressions to the vocabulary elements used by the data, accounting for lexical and structural mismatches (semantic heterogeneity) in doing so, and handling meaning variations introduced by ambiguous and vague expressions, anaphoric expressions, and so on. Authors [50, 51] classify the dimensions of query-dataset lexico-semantic differences (semantic heterogeneity) as:

- Synonym: different lexical expressions mapping to the same concept.
- Lexical differences: Lexical expressions with the same stem or with similar strings mapping to strongly related concepts.
- Conceptual differences: Distinct but related concepts under different lexical expressions in which the alignment satisfies the query information need. It can be Taxonomical differences or Non-taxonomical differences.
- Compositional/Predication Differences: Information may be expressed as different compositions of different dataset elements or predicate structures.
- Functional differences: Aggregated information may be already conceptualized in the database or may need to be computed based on existing data.
- Convention differences: Consists of differences in the conceptualization of the values and units used, numerical vs. non-numerical, dates, dimension, units of measure and scale differences (units of measure, volume, weight, size, currency,
- Null Mappings: Consists of a null mapping from a query term to a database element or vice-versa.
- Intentional Differences: Consists of different intentional definitions expressed by the same term. An intentional definition consists of the properties that a term must satisfy.

One of the causes for meaning variation brings in is ambiguity; a natural language expression can have more than one meaning, map to more than one vocabulary element in the dataset. A more extreme form of ambiguity arises from semantically light expressions, such as the verbs to be and to have, and prepositions like of, with, etc. What vocabulary element they need to be mapped to strongly depends on the linguistic context they occur in [4].

Merely resolving semantic heterogeneity and meaning variation is not sufficient to produce a logical representation that can be used to query a data source. The remaining problem is to determine the overall logical structure of the NL input. This problem

becomes difficult for longer, more complex sentences, where different linguistic phenomena, such as coordination and co-reference, must be handled.

The most common approaches to address the lexical/semantic approximation problems stated above in its many instances, including schema-agnostic queries are the following:

Linguistic resources such as WordNet and Ontologies: these resources are important to compute similarity and relatedness measures, to lookup synonym/hypernym/hyponym, to perform ontology navigation, and to achieve taxonomical and logical inferences to address semantic heterogeneity. Gabrilovich and Markovitch [83] stated the following limitations of this approach:

- Limited number of concepts and relations:
  - ✓ Lack of representation of non-taxonomic relations: WordNet concentrates on the representation of taxonomical and synonymic relations, limiting the computation of semantic relatedness measures.
  - ✓ Meaning evolution: The use of words and their associated meaning is continuously evolving with new contexts of use. WordNet provides a snapshot of consensual meaning descriptions at a certain point in time. The evolution of the meaning of a word requires WordNet to be updated.
  - ✓ Meaning variation in more specific contexts: Meaning can strongly vary in the context in which it is used. WordNet represents common and consensual senses that a word can be used: more specific or particular senses are not covered.
- High construction effort/Low transportability: WordNet was manually created by linguists. There is a high associated cost for updating WordNet to cope with meaning evolution, to cope with other languages or to cover domain-specific scenarios.

Distributional semantic approaches: Most current distributional semantic models exploit the distributional hypothesis: the idea that the context surrounding a given word in a text provides important information about its meaning (means words occurring in similar contexts have similar meanings). The distributional hypothesis assumes that the local context in which a term occurs can serve as discriminative semantic features which represent the meaning of the term [61]. Distributional semantics focuses on the construction of a semantic representation of a word based on the statistical distribution of

word co-occurrence in unstructured data; the meaning of a word is represented by a weighted vector, which can be automatically built from contextual co-occurrence information in unstructured data.

Commonly, there are two types of DSMs; count based and predictive based DSMs. Count based models (i.e. LSA and ESA are the two well know count-based DSM models) are traditional models where they initialize vectors with co-occurrence counts, where similar terms have same counts for different documents. The dimensions of this count matrix are reduced using SVD. Whereas predictive DSMs (e.g. word2vec and glove) also known as embeddings or neural language models are trying to predict (i.e. not count rather predict) surrounding words.

Baroni et al. [64], demonstrated that, in nearly all tasks, predict models consistently outperform count models, and therefore provided us with a comprehensive verification for the superiority of neural word embedding models. Altszyler et al [7], studied and found that Word2vec has outperforming another word-embedding technique (LSA) when it is trained with medium to large corpus size (more than 10 million words)

Tomas Mikolov et al. [3] also noticed that embedding vectors created using the Word2vec algorithm have many advantages compared to earlier algorithms such as Latent semantic analysis, and recommend either of two model architectures of Word2Vec to produce a high-quality distributed representation of words: continuous bag-of-words (CBOW) or continuous skip-gram. In the continuous bag-of-words architecture, the model predicts the current word from a window of surrounding context words. The order of context words does not influence prediction. In contrast, continuous skip-gram architecture uses the current word to predict the surrounding window of context words. The skip-gram architecture weighs nearby context words more heavily than more distant context words. According to the author, Skip-gram model is an efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships and is robust, fast and cheap to train.

b) Dataset Challenges

Data Quality: Strong requirements on question answering systems are completeness and accuracy, "*wrong answers are often worse than no answers*". In the context of linked data, this especially requires a system to deal with heterogeneous and imperfect data.

First, the knowledge available on the linked data cloud is incomplete, so question answering systems should be able to detect when the queried data sources do not contain the answer. Second, the datasets sometimes contain duplicate information, with different datasets using different vocabularies even when talking about the same things. Possibly, datasets also contain conflicting information and inconsistencies, which have to be dealt with.

Distributed Heterogeneous Datasets: The decentralize architecture of the Web has produced a wealth of knowledge distributed across different data sources and different data formats. The challenges imposed by the distributed nature of the Web are, on the one hand, finding the right sources that can answer user query and, on the other hand, integrating partial answers found from different sources. Source selection is one of the challenges in federated question answering approaches.

Indexing Heterogeneous Datasets: A typical QA system is empirically only as good as the performance of its indexing module [41]. The performance of indexing serves as an upper bound to the overall output of the QA system since it can process only as much data as is being presented/served to it from the indices. Most of the systems dealing with unstructured or semi-structured data make use of inverted indices and lists for indexing. For structured datasets, a variety of data structures such as AVL trees, B-Trees, sparse indices, IR trees, etc., have been developed in the past decades [38].

2.7.3 Approaches to Question Answering over Linked Data

Approaches to question answering over linked data vary in several aspects, e.g. being domain-specific or schema agnostic, relying on deep linguistic analysis, on statistical methods, or on graph exploration algorithms, and involving different resources [20]:

a) Approaches based on controlled natural languages**:** typically consider a well-defined restricted subset of natural language that can be unambiguously interpreted by a given system

b) Approaches based on formal grammars**:** typically rely on linguistic grammars that assign a syntactic and semantic representation to lexical units, and exploit the principle of compositional semantics to compute an overall semantic representation of a question by combining the meaning of the parts as specified in the grammar. The benefits of such a system are that, if the corresponding constructs are covered by the grammar, they can deal with questions of arbitrary

complexity. The clear drawback is their brittleness, as they fail if a certain question cannot be parsed by the grammar.

c) Approaches based on mapping linguistic structures to ontology-compliant semantic structures: typically adopt a shallower strategy and try to directly match linguistic structures to semantic triples. To this end, they rely on a measure of similarity that determines the similarity between elements in the query and predicates, subjects or objects in the knowledge base, more or less aiming at computing a mapping between the elements of the query and resources or predicates. While such approaches are more robust than approaches based on formal grammars, they suffer from the fact that there needs to be a one-to-one correspondence between the syntactic structure of the question and the semantic representation.

d) Template-based approaches**:** they typically implement a two-stage process for transforming a natural language question into a SPARQL query. First, they construct a template (or pseudo-query) on the basis of a linguistic analysis of the input question. Second, this template is instantiated by matching the natural language expressions occurring in the question with elements from the queried dataset. The main weakness of such approaches is that often the templates closely correspond to the linguistic structure of the question, thus failing in cases of structural mismatches between natural language and the dataset. While structural variations can be included in the templates, this usually explodes the number of possible queries to build and score.

e) Graph exploration approaches**:** interpret a natural language question by mapping elements of the question to entities from the knowledge base and then proceeding from these pivot elements to navigate the graph, seeking to connect the entities to yield a connected query. The main bottleneck of graph-based approaches is that an exhaustive search of the graph is unfeasible, so that approaches typically explore the graph up to a certain depth or implement some heuristics to make the search over the graph more efficient. Another drawback of graph-based approaches is that more complex queries involving aggregation cannot be answered as they do not have a one-to-one correspondence to a path in the graph.

The above approaches have limitations and strong sides with respect to the solutions they proposed for the above-mentioned challenges. Approaches based on mapping linguistic

structures to ontology-compliant semantic structures (template free approaches) can locate and integrate information from different, heterogeneous semantic resources, relying on query disambiguation, and ranking and fusion of answers and their query approach can scale to large datasets in query execution. But, they adopt a shallower strategy that maps a natural language question to a triple-based representation (semantic triples), the original semantic structure of the question cannot be faithfully captured using those triples. Thus, the systems fail on answering more expressive queries (questions containing Quantifiers like the most and more than, comparatives like higher than and superlatives like the highest).

On the other hand, Template-based approaches that rely on a parse (using a deeper linguistic analysis) of the questions to produce templates that directly mirrors the internal structure of the questions. Accordingly, the systems can answer more expressive queries in contrast to approaches that map natural language input to purely triple-based representations. But the templates are generated manually and domain-specific based, they will incapable to answer questions in open scenario (limited coverage). The approach will have good performance (good precision and recall) for all question's types (including more expressive questions) in contrast to other approaches if the template generation technique improved.

For instance, Template based Question Answering over RDF Data is a semantic question answering system which is based on templates that can reflect the internal structure of the question. For example, consider question (1a) below. PowerAqua [55] would produce the triple presentation in (1b). The goal, however, would be SPARQL query like in (1c). Such SPARQL query is difficult to construct on the basis of the above mentioned triple representations, as aggregation and filter constructs arising from the use of specific quantifiers are not faithfully captured. What would be needed instead is a representation of the information requires that is much closer to the semantic structure of the original question (what Template based Question Answering over RDF Data did).

1. a. Which cities have more than three universities?
   b. ([cities]; more than; universities three)
   c. select count (?A) where {
      ?x rdf: type onto:University.
      ?A onto:city ?z. }
   HAVING (COUNT (?x) > 3)

2.8 Amharic Language

In this section, the literatures written on the Amharic language that are relevant for this thesis are briefly discussed. The historical details about the language, the alphabets, the punctuations, the numbers, and the Morphological complexity, the syntactic and the grammar structure of the language are discussed. In addition to these, the possible challenges that may occur while developing an automated system which involves representing the Amharic language in the machine are indicated.

2.8.1 History about the Language

According to A. A. Argaw and L. Asker [85], Amharic is one of the Semitic languages spoken in north-central Ethiopia. Next, to Arabic, it is the second most spoken Semitic language in the world and it is the official working language of the Federal Democratic Republic of Ethiopia. It is also the native language of perhaps several million Ethiopian immigrants, especially in North America and Israel. It is the second largest language in Ethiopia and possibly one of the five largest languages on the African continent. As a result, it has an official status and used nationwide. Despite it has a large speaker population, the language has little computational linguistic resources.

2.8.2 Amharic Orthography

Unlike Arabic, Hebrew, and Syria (other Semitic languages), Amharic is written using a syllabic writing system, one originally developed for the extinct Ethiopian Semitic language Ge'ez and later extended for Amharic and other Ethiopian Semitic languages. As in other Abugida systems, each character of the Ge'ez (or Ethiopic) writing system gets its basic shape from the consonant of the syllable, and the vowel is represented through more or less systematic modifications of these basic shapes.

The alphabet of the Amharic language consists of 33 core symbols or Fidel (ፊደል). Each of these core symbols occurs in seven different orders; the basic character plus six different symbols or orders formed from the basic character. There are also 37 further characters representing labialized variants of the consonants followed by particular vowels. The complete system has 268 characters. There is also a set of Ge'ez numerals, but nowadays these tend to be replaced by the Hindu-Arabic numerals used in European languages [42].

2.8.3 Amharic Morphosyntax

Amharic Morphology

Amharic is one of the most morphologically complex languages. Amharic nouns and adjectives are marked for any combination of number, definiteness, gender, and case. Moreover, they are affixed with prepositions. For example, from the noun ተማሪ (tämari/student), the following words are generated through inflection and affixation: ተማሪዎች (tämariwoč/students), ተማሪው (tämariw/ the student {masculine}/his student), ተማሪየ (tämariyän/my student), ተማሪየን (tämariyän/my student {objective case}), ተማሪሽ (tämariš/your {feminine} student), ለተማሪ (lätämari/for student), ከተማሪ (kätämari/ from student), etc.

Amharic verb inflections and derivations are even more complex than those of nouns and adjectives [43], consisting of a stem and up to four prefixes and four suffixes. The stem, in turn, is composed of a root, representing the purely lexical component of the verb, and a template, consisting of slots for the root segments and for the vowels (and sometimes consonants) that are inserted around and between these segments. The template represents tense, aspect, mood, and one of a small set of derivational categories: passive-reflexive, transitive, causative, iterative, reciprocal, and causative reciprocal.

Amharic verbs are marked for any combination of person, gender, number, case, tense/aspect, and mood resulting in thousands of words from a single verbal root. As a result, a single word may represent a complete sentence constructed with subject, verb, and object. For example, ይወስደኛል (*yĭwäsdäňal*/[he/it] will take me) is a sentence where the verbal stem ወስድ (*wäsd*/ will take) is marked for various grammatical functions as shown in Figure 2-1.

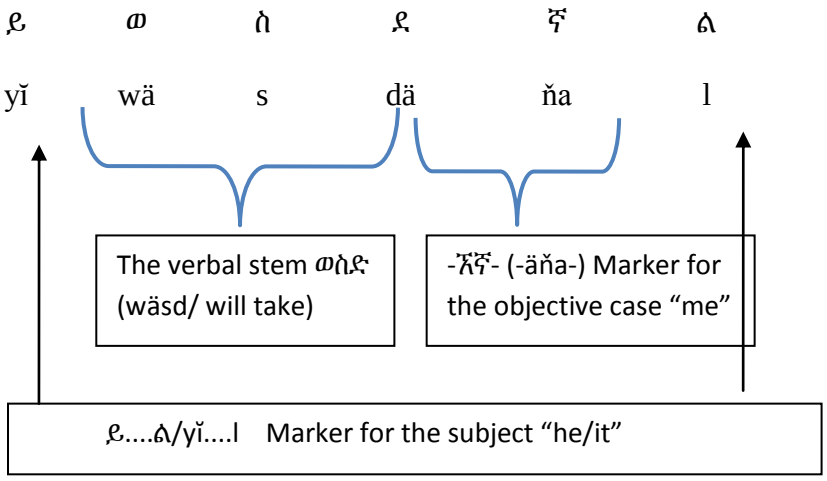

Figure 2-1: Morphology of the word ይመስደኛል

Syntactic Structure of Amharic

Noun Phrases: An Amharic noun phrase has an explicit number, case, and definiteness. The accusative suffix appears obligatorily on definite direct objects and optionally on indefinite direct objects. An unusual feature of the language is the placement of the morphemes marking case (either the accusative suffix or one or another prepositional prefix) and definiteness [45, 46]. These are affixed to the noun itself only when it has no modifiers. If the noun has an adjective or relative clause modifier, the morphemes are normally affixed to the first of these.". Headless noun phrases are common. These consist of one or more relative clauses and adjectives. Examples: tilliqun "the big one (acc.)", yägäzzawn "the one (acc.) that he bought".

Clauses: Unlike in other Semitic languages, all Amharic clauses are headed by verbs [44]. The copula, näw (ነው), is a defective verb with only main clause present forms. Its past is filled by the defective verb näbbär (ነበር), which also serves as the past of the defective verb of existence allä (አለ). In other cases, the copula is replaced by the perfect, imperfect, jussive-imperative, or gerund of either the verb norä "live" or the verb honä "become".

The basic word order of all Ethiopian Semitic languages is subject-object-verb (SOV), a feature that probably results from contact with Cushitic languages. As is common in SOV languages, the order of subject, object, and oblique arguments of the verb is somewhat flexible. In particular, for pragmatic reasons the subject can follow another argument: yohann1s mäskotun säbbäräw, mäskotun yohannis säbbäräw, "Yohannis broke the window". As in other Semitic languages, verbs agree with their subjects in person, number, and (in second and third person singular) gender. Verbs also agree with definite direct or indirect objects, but not both.

As in other Semitic languages, pronoun subjects and pronoun objects are omitted unless they are emphasized. This fact, in combination with the elaborate derivational and inflectional verb morphology, means that sentences consisting of a verb alone or the main verb and an auxiliary verb are not uncommon: alt'äyyäqnatim "we didn't visit her", laflallaˇcihu "shall I boil (sth.) for you please?", awwaddädun "they made us like each other".

Main clause verbs are either in the perfect or a compound imperfect formed from the simple imperfect plus conjugated suffix forms of the defective verb of existence allä. Subordinate clause verbs are in the perfect, simple imperfect, or gerund. tifäll1giyalläš "you (fem.sing.) want", bitt1fälligi "if you (fem.sing.) want".

Cleft constructions are very common in Amharic [47]. Indeed, for questions, cleft constructions are probably more common that non-cleft constructions. In a cleft sentence, the focused argument is placed first, followed by the conjugated copula, followed by other arguments of the original verb, followed by the verb in the relative form: mindin näw yäsäbbäräw "what did he break?" lit. "What is it that he broke it".

Relative clauses in Amharic consist of a relative verb and zero or more arguments and modifiers of the verb, as in any clause. A relative verb is a verb in either the imperfective or perfective with a prefix indicating relativism. As with the main clause verb, a relative verb must agree with its subject and may agree with its direct object if it has one. Both subjects and objects can be relativized: yemiwedat sEt "the woman that he likes".

As noted above, when a noun is modified by a relative clause and has no preceding determiner, it is the relative clause that takes suffixes indicating definiteness or accusative case or prepositional prefixes: yetemereqew lj wendmE new," ' The boy who graduated is my brother." When a sequence of modifiers precedes a noun, it is the first one that takes the suffixes or prefixes: yetemereqew gwebez lj, 'the clever boy who graduated'.

Relative verbs agree with the main clause verbs that contain them. For example, yemiwedat alderesem," (He) who likes her didn't arrive", the third person singular masculine subject in the main clause verb agrees with the third person singular masculine subject of the relative clause verb.

Adverbial clauses are usually indicated through the use of prefix conjunctions on the relative form of the verb (in which case the initial yä is dropped) or the bare imperfect: silämmisäbräw "because he breaks it", bisäbräw "if he breaks it".

As is common in SOV (Subject-Object-Verb) languages [44], Amharic permits the chaining of a number of clauses together in a single sentence without explicit conjunctions indicating the relationship between the clauses. The usual interpretation is sequentiality. All verbs but the final one appears in the gerund form. The final verb may be perfect, compound imperfect, jussive, or imperative. All of the gerund forms agree

with the subject of the final verb. Example: bet tämälliso rat bälto täññа “He returned home, ate dinner, and went to bed” lit. “returning home (3 pers.sing.masc.), eating (3 pers.sing.masc.) dinner, he went to bed”.

2.8.4 Grammatical Rules of Amharic

In addition, since Amharic is morphologically complex language where verbs, nouns, and adjectives are marked for various grammatical functions, the following agreements are required to be checked: adjective-noun, adjective-verb, subject-verb, object-verb, and adverb-verb (Amare, 2010) [15].

Word Sequence: the Amharic language follows subject-object-verb (SOV) grammatical pattern as opposed to, for example, the English language which has SVO sequence of words. For instance, the Amharic equivalent of the sentence “John ate bread” is written as “ጆን (jon/John) ዳቦ (dabo/bread) በላ (bäla/ate)”. Here, the part-of-speech (POS) tags of individual words are used as inputs to check the validity of grammatical patterns.

Adjective-Noun Agreement: Amharic nouns are required to agree for number of modifying adjectives. For example, ረጃጅም ልጆች (räjajĭm lĭjoč/ tall {plural} children) is a valid noun phrase whereas ረጃጅም ልጅ (räjajĭm lĭj/ tall {plural} child) is an invalid noun phrase construction.

Subject-Verb Agreement: Amharic verbs are marked for number, person and gender of subjects. As in other Semitic languages, the subject agreement is expressed by suffixes alone in perfective and gerundive, and by a combination of prefixes and suffixes in imperfective and jussive/ imperative. For example, ልጆቹ መስኮት ሰበሩ (lĭjoču mascot säbäru/the children broke a window) is a valid Amharic sentence. However, ልጅቷ መስኮት ሰበረ (lĭjtwa mäskot säbärä/the girl broke {masculine} a window) is not a valid Amharic sentence since the subject ልጅቷ (lĭjtwa/the girl) is feminine and the verb ሰበረ (säbärä /broke {masculine}) is marked for masculine. Amharic is a null subject language; that is, a sentence does not require an explicit subject, and personal pronouns appear as subjects only when they are being emphasized for one reason or another.

Object-Verb Agreement: An Amharic verb may also have a suffix representing the person, number, and gender of a direct object or an indirect object that is definite. The corresponding suffixes in other Semitic languages are often considered to be clitics or even pronouns, but there are good reasons not to do so for Amharic. First, one or two

other suffixes may follow the object suffix. Second, as with subjects, object personal pronouns may also appear but only when they are being emphasized.

Adverb-Verb Agreement: Tenses of verbs are required to agree with time adverbs. For example, ትላንት ሰበሩ (*tĭlant säbäru*/ [they] broke yesterday) is a valid verb phrase construction whereas ትላንት ይሰብራሉ (*tĭlant yĭsäbralu*/ [they] will break yesterday) is an invalid construction.

Challenges that may occur while developing an automated system which involves representing the Amharic language in the machine are:

- *Spelling Variations:* In Amharic, there are words which can be written in different formats. For instance, "ፀሐይ" and "ጸሐይ", "ፀሀይ" and "ጸሀይ" all mean the same, "sun" although they are written differently (indicating alternate use of ፀ and ጸ, and ሀ and ሐ. As it is stated in [46], historically these characters have different sounds. In query processing, there should be a mechanism to handle this characteristic of the language.
- *Different ways of writing the same word:* it is very common to see different people writing the same word in a different way. This spelling variation happens mostly when a word from other language is used without translation which is referred to as transliteration. For example, ኮምፅዩተር and ኮምፒዩተር, ፕርምየር ሊግ and ፕርሚየር ሊግ
- *Formation of compound words:* compound words are formed by joining two words with or without modification. For example, the word "ኳስ ሜዳ" is formed from the words "ኳስ" and "ሜዳ" without any change in the words. In another case, for instance, a compound noun ቤተክርስቲያን is formed from two words, ቤት and ክርስቲያን with a little modification on the word ቤት. Moreover, those compound nouns that are formed without modifications can be written with or without a gap between the words like "ኳስ ሜዳ" and "ኳስሜዳ". In spite of this writing variation, with or without a gap, the meaning of the words is the same.
- *Agglutinative Nature of the Language:* Agglutinative property means that some additional features can be attached to the word to add more complex meaning. According to [48], most of the time NE root words cannot be found in a document, rather they are found in their morphological derivations that are formed by affixes.

- *Ambiguity:* is an attribute of any concept, idea, statement or claim whose meaning, intention or interpretation cannot be definitively resolved according to a rule or process consisting of a finite number of steps. In order to understand Amharic texts (in our case the query), there is a need for resolving ambiguity at a different level. Getahun [46] identified six types of ambiguity in Amharic: Lexical Ambiguity, Phonological Ambiguity, Structural Ambiguity, Referential Ambiguity, Semantic Ambiguity, and Orthographic ambiguity.

Chapter 3 - Related Work

In this Section, related researches conducted both for Amharic and non-Amharic languages are discussed.

3.1 Amharic Search Engine Systems

Different researchers have tried to build search engine/ question answering to fill information need of a user from large collections Amharic text. Each of these research works has its own strong and weak parts.

Teteyeq: Amharic Factoid Question Answering System

The researches in [9] implemented handwritten rules in question classification and answer extraction component of the QA system. The named entity (gazetteer) and pattern-based answer pinpointing algorithms have been developed in order to locate possible answer particles in a document. The overall system has four stages. In the first stage, the question processing component classifies questions to appropriate question types, determines the expected answer type and generates proper information retrieval query. The second stage, the document retrieval component retrieves relevant documents that will be further processed by the later part. The third is sentence/paragraph ranker that re-ranks sentences and paragraphs based on the relevance of the candidate answer particle of the sentence and paragraph. The final component, the answer selection component, selects five top-ranked answers for the users.

The researchers have developed algorithms for the different components of the system, and tested the performance of them and got promising results. But the indexing component of the system is done using lucent indexer. Lucent uses the IDF (Inverse document frequency) approach for indexing. As IDF is a keyword-based method, the words (i.e., those words which are used to represent documents) may not be capable of reflecting the semantics of the documents and can't to capture the implicit relation among them or the semantics of words in the document. If the question and the index have no word in common, then the system will respond no result to the user (low recall). Besides, it will retrieve irrelevant document to the user (low precision).

Towards Amharic Semantic Search Engine

This research was conducted to build a semantic search engine, AmhS2Eng, which returns a list of Amharic documents very similar to the user query either directly or

indirectly [5]. Documents extracted from the Web and user queries are annotated with semantic concepts extracted from the football domain. It is composed of four major components: 1) The Ontology is built manually with the help of domain and language experts. It contains concepts, the relationship between concepts and instances of each concept and similarity between knowledge entries. 2) The crawler extracts URLs and content of Amharic document and stores them in a document repository. 3) The indexing component is responsible to annotate the crawled document with semantic information (Concept, Instance, Words) and associate semantic information with weight.4) The query processor responsible to annotate the original user query with semantic information, conduct document retrieval, rank the resulting documents based on relevance as explained in the next sub-sections.

The researcher has tested the system based on the relevant information provided by domain experts, and 138 football news articles and 25 queries were used. But this retrieval system has some weakness. The first is that the system was limited to searching for information in a single domain, football domain (i.e. searching on only one kind of information). But there are user's queries which are much complex and need integration of information from different related data sources. As it was work on a specific knowledge base (i.e. closed domain), it did not need to integrate different schemas and knowledge bases and thus suffer less from ambiguity. However, it lacks flexibility and requires high costs when adopting such a system to a new domain or implementing a new system. The second is that the system is Ontology-based information retrieval: It annotates and index documents, and expands user query using background ontology. It uses background knowledge to enhance retrieval accuracy. However, they did not utilize the background knowledge for semantically answering user queries. By improving query processing, and answer extraction and retrieval component of the system, we plan semantic searching on the open domain. The other one is that the system had lack of delivering readymade (or direct) information to the user, instead it retrieved documents contain the needed information, which the user must then scan to search for pertinent information.

3.2 Search/Query Engine for Other Languages

3.2.1 Template based Approaches

CASIA: A Question Answering System over Linked Data

This research [13] was conducted to build a question answering system over Linked Data, which translate the natural language questions into the standard RDF data queries (SPARQL). The pipeline of the system consists of three main components, including question analysis, resource mapping, and SPARQL generation. In specific, they used shallow (POS tagging, NER), deep linguistic analysis (dependency parsing), and question type analysis to transform each natural language question into query triple (graph pattern template). Their main heuristic to construct query triple from parsing tree was extracting subject-predicate-object and proposition phrases relations. Then they used PATTY, WordNet and other free external resources for mapping the phrases in the query triple to the right resource in linked data to generate SPARQL query. At last, based on the identified question type, they generated the SPARQL queries and used the DBpedia endpoint to validate them.

The researchers have evaluated the system based on the QALD-3 test dataset and achieved an F-measure score of 0.36, an average precision of 0.35 and an average recall of 0.36 over 99 questions. But the system has some weakness. The first one is that the system cannot address questions which contain comparative, superlatives and negation words. The second one is that the system cannot answer complex questions with relative sentences and conjunctions. This is because the linguistic analysis of the system cannot identify all relations (i.e. wrong relation mapping) in questions. The second problem is that they did not resolve the entity disambiguation (i.e. entity linking) problems; some entities extracted from question are wrong using string matching. In addition, their heuristic to generate query triple is not sufficient and generates wrong query triples for some questions. The other problem is they cannot detect the right order between subject and object.

TBSL: Template-based Question Answering over RDF Data

TBSL [54] is a question answering system that focuses on transforming natural language questions into SPARQL queries in such a way that the query is a faithful representation of the semantic structure of the question. To this end, TBSL first produces a SPARQL

template that mirrors the internal structure of the question and that is then instantiated with URIs by means of statistical entity identification and predicate detection. For parsing user question, they rely on developing a lexicon, which used to specify for each expression a syntactic (i.e. trees from Lexicalized Tree Adjoining Grammar) and a semantic representation (i.e. Underspecified Discourse Representation Theory). Such a lexicon consists of two parts. One part comprises domain-independent expressions, which were specified manually and can be re-used across all domains. The other part of the lexicon comprises domain-dependent expressions which are built on-the-fly on the basis of part-of-speech information, and a set of simple heuristics that specify which POS tag corresponds to which syntactic and semantic properties.

The general workflow of the system is: The input question, formulated by the user in natural language, is first processed by a POS tagger. Then both domain dependent and independent lexicon are used for parsing, which leads to a semantic representation of the question, which is then converted into a SPARQL query template. The templates contain slots, which are missing elements of the question that have to be filled on-the-fly during parsing. To fill them, they first generate natural language expression based on a set of heuristics for possible slot filler from the user question using WordNet expansion (i.e. synonym). In a next step, they used approaches relying on identifying and retrieving the most common synonym set (or synset) from WordNet and computing string similarity for entity identification, and using of the pattern library extracted by the BOA (Bootstrapping Linked Data, is a framework to compute natural language expressions that stand for predicates from an arbitrary input corpus) framework in addition to string matching to detect properties. This process yields a range of different query candidates as potential translations of the input question. Thus they combined string similarity values, prominence values, and schema conformance checks to rank those query candidates

The researchers have evaluated the system based on QALD-1 benchmark on DBpedia. It comprises two sets of 50 questions over DBpedia, annotated with SPARQL queries and answers. Each question is evaluated with respect to precision (number of correct resources returned by system/number of resources returned by the system) and recall (number of correct resources returned by system/number of resources in the gold standard answer). Based on the evaluation they achieved a precision of 0.61, a recall of 0.63 and F-measure of 0.62. The main strength of their approach is that the generated

SPARQL templates can capture the semantic structure of the natural language query. Thus, questions containing quantifiers like the most and more than, comparative like higher than and superlatives like the highest can be answered by the system. However, as the writer said, "in some cases, the semantic structure of the question and the triple structure of the query do not coincide, thus faithfully capturing the semantic structure of the input question sometimes leads to too rigid templates". They suggested two approaches to solve the problem. The first one concentrates on more flexible processing; the relaxation of templates, such that the triple structure is not completely fixed but is discovered through exploration of the RDF data. The second approach concerns incorporating a more flexible fallback strategy in case no successful SPARQL query is found; combining active learning method that allows the user to give feedback on the presented query result. This system uses regularities in how syntactic patterns map to semantic ones (i.e. their heuristics) to create templates.

The linguistic analysis component, pattern library component, domain-independent expression, and rules used to generate domain-dependent lexicon are language dependents which support only the English language. Beside its language dependency, this system has some weakness. The first drawback of this approach is the limited coverage of templates, making them brittle when it comes to unconventional question formulations. Their approach does not support complex questions that require multiple KG predicates to obtain an answer. For example, the question "Which were the alma maters of the PR managers of Hillary Clinton?" is answered by a chain of KG predicates. The other drawback is that the approach doesn't incorporate inference techniques for semantically light expression (e.g. preposition), this leads the hard case for entity identification when a property in the intended target query does not have a correspondent in the natural language question. And their query-dataset semantic matching strategies strongly rely on WordNet which is restricted to taxonomic/synonymic relations, but there are cases which require more exploration for semantic approximation (or semantic relatedness measure), hard to find a match using WordNet relation only. So, these cases would require the incorporation of additional semantic similarity measures.

3.2.2 Template Free Approaches based on Mapping Linguistic Structures to Ontology-Compliant Semantic Structures

PowerAqua: Supporting Users in Querying and Exploring the Semantic Web

PowerAqua [55] is a question answering system focusing on querying multiple datasets (ontologies) on the Semantic Web, that is able to answer queries by locating and integrating information, which can be distributed across heterogeneous semantic resources. To this purpose, PowerAqua supports query disambiguation, knowledge fusion (to aggregate similar or partial answers), and ranking mechanisms, to identify the most accurate answers to queries. The input for the system is based on natural language queries. To do so, PowerAqua follows the pipeline architecture: The first step consists in a linguistic analysis of the question using GATE natural language processing tool, in order to detect the question type and to translate the natural language question into a triple-based representation, into so-called query triples (is defined by the following structure: <subject, predicate, object>).

The second step, The Element Mapping Component, then searches for candidate instantiations of the occurring terms. This involves detecting ontologies on the Semantic Web that are likely to contain the information requested in the question and finding vocabulary elements that match the terms. The mapping process for crossing the semantic gap between user queries and dataset are concentrated in the PowerMap component (i.e. Element Mapping component), consists of three phases. The first phase has the objective of identifying candidate mappings for the query terms in different ontologies. The matching is based on a string similarity approach, using edit distance and WordNet to lookup synonyms and hyponyms. From the set of ontologies and mappings identified in the first phase, PowerMap excludes terms which are not semantically consistent, exploring the ontology hierarchies to determine the correct sense of the word and using WordNet-based semantic similarity measures to map between query terms and ontology classes. The third phase has the objective of identifying mappings that better represent the query domain determining the ontologies which better cover the query domain. The output is a set of tables containing matching semantic elements for each term occurring in the query triples. These mappings can be used to turn query triples into ontology triples.

Next, the relation similarity service (RSS) component determines the most likely interpretation both of the terms in the query triples and the question as a whole, on the basis of the linguistic analysis of the question and the ontological context. This can also involve modifying query triples, e.g. by splitting compound terms. Finally, given a ranked list of ontology triples, answers are retrieved from the involved data sources. Since these can be multiple different sources, this step also involves identifying semantically equivalent or overlapping information in the answer set, in order to avoid duplicate results, a ranking of answers, as well as integrating answer fragments from different sources.

The researchers have conducted six main evaluations to test the system. The first type of evaluation focuses on assessing whether the exploitation of PowerAqua as a query expansion module, provided an improvement in precision over a keyword-based retrieval baseline. To represent the semantic information space, they collected and indexed in the PowerAqua's storage platforms around 2GB of metadata comprising different domains. The system successfully generated a query expansion for 20% of the queries. The second type of evaluation focuses on the analysis of the capability of the system to answer queries using the information provided by multiple datasets defined by multiple vocabularies. The evaluation primarily assesses the ability of the system to map a user query into ontology triples in real time. A dataset of 69 questions was built manually by asking volunteers to build questions based on data available in one or more datasets. Each question is self-contained with no references to previous queries. Just precision is measured. The ontologies are based on datasets such as SWETO DBLP or SWETO (approximately 800.000 entities and 1.600.000 relations). 2GBs of data stored in 130 sesame repositories were collected. In this evaluation, the authors report an average precision of 69.5% of the queries with an average execution time per query of 15 seconds.

In a second evaluation, PowerAqua is also evaluated using the Question Answering over Linked Data (QALD-1) test collection, where precision and recall are collected over a set of 50 natural language queries over DBpedia. The schema size of DBpedia, containing thousands of open domain predicates and millions of instances provides a suitable dataset for the evaluation of schema-agnostic scenario. Queries within the QALD test collection explore complex natural language query patterns. The third type of evaluation focuses on the analysis of merging and ranking capability of the system. A dataset of

total of 40 questions was collected and selected from the example queries in the PowerAqua demo website and from the previous PowerAqua evaluation. As judgments to evaluate the merging and ranking algorithms, the researchers prepare two ontology engineers and then they provided a True/False manual evaluation of answers for each query. Precision and recall were selected as evaluation metrics.

PowerAqua's main strength is that it locates and integrates information from different, heterogeneous semantic resources, relying on query disambiguation, and ranking and fusion of answers. The linguistic analysis component is language dependent which supports only the English language. Beside its language dependency, this system has some drawbacks. The linguistic component of the system is based on a mapping of a natural language question to a triple-based representation. But, the semantic structure of the question cannot be faithfully captured by using this representation, thus complex questions containing quantifiers, comparatives, and superlatives pose a problem. This is its main weakness. The other weakness is that, the mapping mechanism which is the element responsible for matching the query terms to vocabulary terms relies on WordNet and taxonomical information within the ontologies for computing semantic approximations. In relation to the vocabulary gap, many query-dataset mappings go beyond synonymic or taxonomic relations as we have discussed in Chapter two, this provides an incomplete solution for the semantic matching problem.

3.2.3 Graph Exploration Approaches

Treo: Schema-Agnostic Querying using Distributional Semantics Approach

Treo [56] is question answering system focus on addressing the vocabulary mismatch problem which is central for schema-agnostic querying over large schema and heterogeneous linked datasets. To address this problem the approach used the distributional semantic hypothesis to automatically construct a semantic model based on the statistical distribution of word co-occurrence in texts, allowing the creation of an associational and quantitative model which captures the degree of semantic relatedness between words.

The general workflow of the system is: The first step is constructing of a distributional semantic model based on the extraction of co-occurrence patterns from large corpora (e.g. Wikipedia), which defines a distributional semantic vector space (i.e. uses concept vectors to semantically represent data and queries, by mapping datasets elements and

query terms to concepts in the distributional space). In the second step, the RDF graph data is embedded into space, defining the T − Space, a structured distributional semantic vector space (i.e. indexing Linked data into T − Space), built using the Explicit Semantic Analysis (ESA) as the distributional model. It used as large-scale common-sense information embedded in the distributional model to be used in the semantic matching/approximation process. The above two steps are made offline to index-linked data into T − Space; it is ready to be queried.

The question analysis step (step 3) consists in recognizing and classifying entities and operations in the question, as well as in mapping the natural language question into a partial and ordered dependency structure (PODS), a triple-like pattern, and a set of query features. The analysis consists of the following specific operations: parsing of the question according to its dependency structure and the occurring parts of speech, detection of the query focus and answer type based on rules over part-of-speech tags, determine the type of detected entity candidates (instances, classes and properties) using part-of-speech tag pattern rules, detect operations and associated parameters in the question using part-of-speech tags and keyword patterns, reduces the dependency structure constructed when parsing the input question to a set of PODS by applying two sets of operations: the removal of stop words and their associated dependencies, and the re-ordering of the dependencies based on the core entity position in the query, and classifies the query according to query features that represent database primitives on the schema level (instances, properties, classes), on the operational level (e.g. aggregation and ordering), and on the structural level (conjunction, disjunction, property composition).

After the query is analyzed, a query processing plan is generated, which maps the set of features and the semi-structured query into a set of search, navigation and transformation operations (step 4) over the data graph embedded in the T − Space; Search operations consist of keyword and distributional search operations over the data graph in the T − Space, in particular, instance term search (over a term space), distributional class search, and property search. Graph navigation and transformation operations consist of graph composition and on the application of order, user feedback, aggregation, and conditional operators.

The researchers have conducted an evaluation using the Question Answering over Linked Data 2011 challenge test collection (QALD-1). The query set contains 76 natural

language queries over DBpedia 3.6 containing rdf:type links to YAGO classes. The dataset was indexed into the $\tau -$ Space. The Easy-ESA distributional infrastructure based on Wikipedia 2006 was used.

They used five metrics to measures the relevance of the results returned by the schema-agnostic query mechanism: % of queries answered, precision, recall, f-measure, mean reciprocal rank (MRR) for each query and the averages for the whole query set. In this evaluation, the authors report an average precision of 0.63, an average recall of 0.79, an average F-Measure of 0.70, an average of MRR of 0.49, % of queries answered 0.79. Treo's main strength is that it provides a more robust semantic matching that addresses previous limitations of WordNet-based semantic matching. The linguistic analysis component is language dependents which supports only the English language. Beside its language dependency, this system has some drawback. Treo's approach combines entity search, spreading activation search, and semantic relatedness to navigate over the linked data Web graph. Thus, its main drawback is that more complex queries involving aggregation cannot be answered as they do not have a one-to-one correspondence to a path in the graph.

3.2.4 Approaches based on Formal Grammars

Pythia: Ontology-Specific Question Answering

Pythia [57] is the ontology-based question answering system. This means that ontologies play a central role in interpreting user questions (i.e. user questions are interpreted with respect to an ontology underlying the dataset that is queried). For example, ontological knowledge is used for drawing inferences, e.g. in order to resolve ambiguities or to interpret semantically light expressions. Pythia constructs meaning representations whose vocabulary is already aligned to the vocabulary of the ontology. To this end, they rely on ontology-lexica that make the possible linguistic realizations of ontology concepts in a particular language explicit (i.e. in lemon format, is a model for the declarative specification of multilingual, machine-readable lexica in RDF that capture syntactic and semantic aspects of lexical items relative to some ontology). The meaning of a lexical item is given by reference to an ontology element, i.e. a class, property or individual, thereby ensuring a clean separation between the ontological and lexical layer

The general workflow of the system is: First, the ontology-lexicon is used to automatically construct principled linguistic representations. Pythia builds on

Lexicalized Tree Adjoining Grammars (LTAG) as syntactic formalism and Dependency-based Underspecified Discourse Representation Structures (DUDES) for specifying semantic representations. Those linguistic representations together with domain-independent representations, e.g. for determiners and auxiliary verbs, constitute the grammar (i.e. are constructed offline) that is used for parsing and interpreting an input question (step 2). In particular, the interpretation process is compositional, meaning that the semantic representation of the input is recursively computed on the basis of the meaning of the words in the input as well as the way the words are connected syntactically. Finally, the resulting meaning representations are transformed into formal queries, in particular, F-Logic and SPARQL queries.

Pythia used the Tang & Mooney [24] test collection (865 user questions) to evaluate the quality of the approach and they reached a recall of 67% and precision of 82 %, leading to an F-measure of 73,7%. Its main strength is that it Implemented deep linguistic analysis allows to construct formal queries even for complex natural language questions, e.g. involving quantification and superlatives. And they used a vocabulary that is aligned to the vocabulary of the ontology (i.e. ontology lexicon) therefore offers a very precise and correct way of matching natural expressions with ontology concepts. Pythia thus is domain-specific question answering system that requires an ontology lexicon for the domain that is queried. The main limitation consists in the effort required for building the support lexica, either manually or semi-automatically. And although a grammar-based interpretation process offers high precision, systems relying on domain grammars usually suffer from limited coverage.

3.2.5 Machine Learning Approaches

IBM Watson's DeepQA

The IBM Watson DeepQA system [58] is a question answering system where information sources are both unstructured and structured data. The main components of the systems are: The Question analysis component consists of a mixture of question analysis methods including shallow and deep parsing, extraction of logical forms, semantic role labeling, co-reference resolution, relations extraction, named entity recognition. The Question decomposition component consists of the decomposition of the question into separate phrases, which will generate constraints that need to be satisfied by evidence from the data. The Hypothesis generation component consists of

two steps: primary search and candidate answer generation. The primary search step focuses on finding all content that can support the question (maximizing recall). Different information retrieval techniques for document and passage retrieval are used over unstructured data sources and SPARQL queries are used over triple stores. Triple store queries in the primary search step are based on detected named entities, relations or lexical answer types in the question. The second step consists of the candidate answer generation, where information extraction techniques are applied to the search results to generate candidate answers.

The soft filtering component consists of the application of lightweight (less resource intensive) scoring algorithms to a larger set of initial candidates to prune the list of candidates before the more intensive scoring components. The Hypothesis and evidence scoring component consist of two steps: supporting evidence retrieval and deep evidence scoring. The supporting evidence retrieval step seeks additional evidence for each candidate answer from the data sources while the deep evidence scoring step determines the degree of certainty that the retrieved evidence supports the candidate answers. The system uses more than 50 scoring components that produce scores which range from probabilities and counts to categorical features. Scores are then combined into an overall evidence profile which groups individual features into aggregate evidence dimensions. The Answer merging component consists of merging answer candidates (hypotheses) with different surface forms but with related content, combining their scores. The Ranking and confidence estimation component is the last step, in which the system ranks the hypotheses and estimate their confidence based on the scores, using machine learning approaches over a training set. Multiple trained models cover different question types.

Despite the fact that most of the evidence analysis in the IBM Watson DeepQA system is based on unstructured data, different components of the system rely on structured and semi-structured data and certain question types are answered by structured data sources. In particular, the DeepQA system uses three types of structured data sources:

- ✓ structured data available online (entity and associated types, movie databases)
- ✓ specific structured data extracted from unstructured data
- ✓ curetted data providing additional information to the system (e.g. question and answer types)

3.2.6 Other Approaches

An HMM-based Approach to Question Answering against Linked Data

This research [12] was conducted to build a QA system enabling NL questions against Linked Data. The system integrates lexical semantic modeling and statistical inference within a complex architecture that decomposes the NL interpretation task into a cascade of three different stages: (1) The selection of key ontological information from the question (i.e. predicate, arguments and properties), (2) the location of such salient information in the ontology through the joint disambiguation of the different candidates and (3) the compilation of the final SPARQL query.

They used Hidden Markov Models (HMM) to select the proper ontological triples according to the graph nature of DBpedia. In the first step, the syntactic dependency graph of the question is analyzed to find all grammatical relevant elements (nouns, verbs, and adjectives) and initialize the HMM. This information is used for computing the optimal sequence of states including disambiguation which is finally used to convert the sequence to a SPARQL query. In this way, they solved the localization and retrieval of ontological elements evoked by a question without relying on strict hypothesis on the resource vocabulary, and the different ambiguities arising in the interpretation by integrating question grammatical structures and ontology information. The researcher has evaluated the system based on the QALD-3 test dataset and achieved an F-measure score of 0.33, an average precision of 0.32 and an average recall of 0.34 over 99 questions. The major source of the smaller recall and precision of the system (drawback) was incapable of properly locating relation (predicate) in the question. This is because the system did not resolve anaphoric references, unable to identify implicit arguments (i.e. they basically work on the predicate that has an explicit lexical reference), and sentence segmentation and identification of predicates with their arguments basically dependent on the dependency graph, makes use of no semantics, e.g. no access to the DBpedia resources.

FREyA: An Interactive Way of Querying Linked Data Using Natural Language

FREyA [59] is a question answering system which employs feedback and clarification dialogs to resolve ambiguities and improve the domain lexicon with the help of users. User feedback is used to enrich the semantic matching process by allowing manual

query-vocabulary mappings. The query processing workflow starts with the syntactic parsing and analysis (using the C-structures from the Stanford Parser) (question analysis component). The question analysis components also identify candidate question terms which can be potentially mapped to dataset concepts, using a rules-based approach over the query the C-Structure and POS Tags.

The ontology-based lookup component maps question terms to the dataset entities. If the system fails to generate a mapping between question and dataset, it returns a dialog to the user through the consolidation algorithm component. The consolidation algorithm detects ambiguous concepts (through the disambiguation dialog) or provides alternative mappings based on a relaxation of the query-ontology mapping using neighboring terms (mapping dialog). The mappings are stored by the combination of the question-dataset terms and the surrounding context terms, allowing the improvement of the performance over time. After the mappings are completed, the system detects the answer type and combines the entities identified in the dataset into triples by taking into account the dataset structure

FREyA is evaluated using precision and recall under the Question Answering over Linked Data 2011 test collection (using MusicBrainz and DBpedia datasets with 100 training and test questions) and reached a recall of 0.54 and precision of 0.63, leading to an F-measure of 0.58. The core contribution behind the FREyA system is the exploration of user interaction/feedback element into for disambiguation and semantic enrichment.

3.2.7 Summary

Previous template based question answering systems exhibited as good approaches to capture the semantic structure of the natural language question. Accordingly, the systems can answer more expressive queries containing quantifiers, comparative, and superlative in contrast to approaches that map natural language input to purely triple-based representations. However there are still gaps that require more studies; the first is that the question analysis and parsing are language specifics, and rely on manually built in domain-specific lexicon, manually crafted grammer and linguistic features.

The second is that the query-dataset semantic matching techniques which they used are strongly rely on Wordnet and domain specific Ontologies which they are restricted to taxonomic/synonymic/hyperneym relations, but there are cases which require more exploration for semantic approximations and as full fledged Amharic wordnet is not

available, require long time and language experts to built it manually. The other is that they do not support compex questions that require multiple KG predicates from different data sources to obtain answers.

We propose a novel approach to Amharic Semantic Query answering over Linked Data that relies on a deep linguistic analysis (via neural semantic parsing, learning based) yielding a SPARQL template with slots that directly mirrors the internal structure of Amharic question, need to be filled with URIs. In order to fill those slots, possible datasets were disambiguated, and entities were identified using string similarity as well as semantic relatedness using distributional semantic model developed from text documents. The remaining query candidates were then ranked on the basis of scores attached to the entities and BGPs and finally top ranked query was selected to be executed as the final result.

Chapter 4 - Design of Natural Language Semantic Question Answering

4.1 Overview

In this Chapter, the design and implementation of the proposed Amharic semantic query answering over Web of Data are presented in detail**.** As depicted in *Figure 4-1*, the proposed system architecture is composed of five major components: Word Embedding, Data Indexing, Query Template Generation, Resource Matching and Disambiguation, and Query Ranking and Execution. The Data Embedding part composed of pre-processing of Amharic reference text corpora (such as Normalizing, Stemming, and Stop word removal) and Distributional word embedding model. The result of this component is a vector representation of words, which is used later in Data indexing component.

The major activities during Query Template Generation are Question preprocessing (such as Stemming, Normalization…), Deep neural semantic parser that map preprocessed input question to formal meaning representation (its logical forms), Template generator which is responsible to translate meaning representation to domain-independent SPARQL query templates by using manually curetted Templates, which they are later instantiated using entity matching and disambiguation component.

The Resource Matching and Disambiguation component is responsible for entity matching and resource disambiguation to generate Templates with disambiguated slots. The SPARQL query Templates generated by Query Analysis component contains slots, which are missing elements of the query that have to be filled with disambiguated entities. And our dataset where the query is processed is an interlinked data (collected from different data sources) rather not a single dataset, thus it needs disambiguation of resources and federated query to retrieve answers from disambiguated resources.

To do so, this component composed of three sub-components; entity matching subcomponent, it returns matched candidate entities through term-based matching and distributional semantic searching from indexed data. Those returned candidate entities are used by disambiguation subcomponent to select (rank) important data resources for the query. The rewriting/merge subcomponent uses the selected data resources to rewrite SPARQL query Templates generated from the Template generation component and generate candidates of federate SPARQL query templates with slots, which they are

ranked and processed using Ranking and Answer processing component of the system. Ranking and Answer processing component is a response to rank SPARQ queries, and to retrieve top-ranked answers to the user. Word Embedding and Data Indexing are the two offline components. The other components are an online one.

4.2 Word Embedding

It is responsible to build word embedding model that embrace the vectors representation of words in the given input Amharic text corpus. This component has the following sub-components:

4.2.1 Corpora Preprocessing

As we have discussed at the beginning, the goal of this module is generating indices which are important for computing semantic relatedness between natural language expressions in the query and labels of vocabulary elements in the datasets. As we have reviewed in Chapter two, existing semantic matching approaches currently strongly rely on WordNet and on explicit taxonomical relations (domain-specific ontology) in the data to support lexical/semantic approximation in the dataset. But these approaches have limitations in order not to use in our works; no full-fledged Amharic WordNet (and they consume time and resource to manually prepare) available, can't capture domain-specific semantics, restricted to semantic approximations techniques which strongly rely on taxonomic/synonymic relations, limited in addressing the computation of similarity between terms crossing part-of-speech boundaries or containing multi-word expressions, and inability to capture uncommon and new terms or term senses.

Instead, we rely on a distributional semantic that address the above limitations; it is an approach that automatically encodes large semantic and commonsense knowledge bases from large unstructured text corpus (in our case Amharic text corpus). To do so, we need to first preprocess the Amharic text corpus that increases performance and reduces runtime of the extraction. This process is language dependent and includes the following activities; building a corpus**,** removing unnecessary words, changing characters and words into their common form, and stemming.

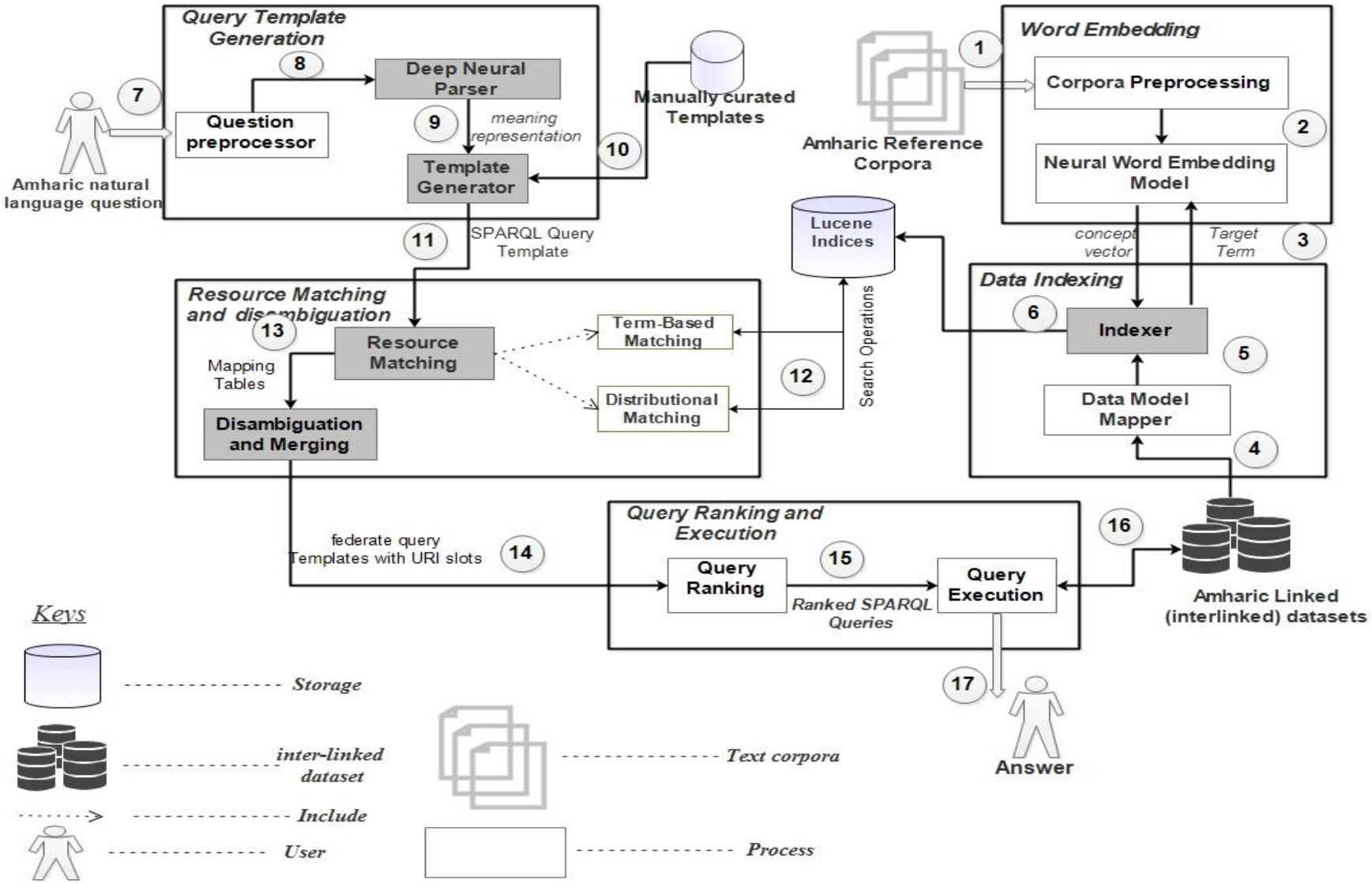

Figure 4-1: Architecture of the proposed system

Building Corpus

We collect and build a corpus by crawls the Web, and from a set of publicly available Amharic text collections. Its' contents are mainly news articles based on a local Amharic newspaper, Amharic editions of international news networks, Amharic version of the bible, and Amharic Wikipedia dump.

Removal of Extraneous Characters

The numbers, punctuation marks and control characters in the text of each file were not considered for building a distributional model as they do not provide important information about target word meaning. Words containing numbers like (2nd i.e. 2ኛ or ኢዜአ103862) were excluded at the first phases of preprocessing. Moreover, the standard control character; Amharic punctuation marks ።፣ ፥ ፦ ፤ ፡- and symbols borrowed from other languages (?, ! , " , ", ', /, \, etc.) were ignored.

Normalization

In Amharic, there are words which can be written in different formats. It would unnecessarily increase the number of words to be indexed which could reduce the efficiency and accuracy of the indexer. So, this activity normalizes these spelling variations by changing the different forms of a character into one common form. The other normalization issue is related with shorthand representation of words like አ/አ, ት/ሚ and ጽ/ቤት. Therefore, these forms should be converted into their expanded long forms. *Table 4-1* shows an example of the character redundancy where more than one symbol is used for the same sound.

Table 4-1: Redundant Amharic characters

Consonants	Other symbols with the same sound
ሀ (hä)	ሃ ሐ ሓ ኃ ኀ ኻ
ሰ (sä)	ሠ
አ (ä)	ኣ ዐ ዓ
ጸ (tsa)	ፀ

The other normalization activity is stemming, the process of reducing inflected (or sometimes derived) words to their stem. In our work, it is sufficient that related words map to the same stem (even if this stem is not in itself a valid root), and it provides a

means to minimize index terms and hence save storage space, increases the performance of indexer (accuracy and efficiency). In this thesis, we employ the stemming algorithm developed in [60]. Normally proper names, dates, and numbers (i.e. resources and values) should not be subjected to stemming since they will not be reduced to root words.

Formation of Compound words

Compound words are used as a single word in some instances (either by fusing the two words or by inserting a hyphen between them) and as two separate words at other instances. Inconsistent usage of compound words could result in redundant word features by creating more words. A compound word (example አዲስ አበባ) can be treated as two separate words አዲስ and አበባ. In order to minimize this difficulty, the popular compound nouns are known in each domain are identified and stored as knowledge to be used appropriately whenever required.

Sub-sampling Frequent Words

It removes most frequently occurring words from the text that do not provide important information about target words meaning. The assumption is that words which occur frequently in almost all text are non-informative. Removal of frequent words during training results in a significant speedup (around 2x - 10x), and improves the accuracy of the representations of less frequent words [63]. We identified two kinds of frequent words: *Common Stop Words*: Like other languages, some words in Amharic are used very frequently in the normal usage of the language. The other is *Amharic news specific words*: as our data is more collected from Amharic news, there are vocabularies that are peculiar and frequently used by reporters and journalists. Thus, we collected very frequent words by randomly picking words (wi) whose frequency $f(w_i)$ is higher than threshold t=$1 \times e^{-5}$ (Authors [64] suggest $1 \times e^{-3} - 1 \times e^{-5}$ as a useful range) with a probability (p(w_i)> 0.9999) of computed by the formula:

$$p(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \qquad (4.1)$$

We used this sub-sampling formula because it aggressively subsamples words whose frequency is greater than t while preserving the ranking of the frequencies. We final aggregated both kind of frequent words and removed them from the collected corpus.

4.2.2 Neural Word Embedding

As we have reviewed in Chapter two, most current distributional semantic models exploit the distributional hypothesis; the idea that the context surrounding a given word in a text provides important information about its meaning. This sub-component focuses on the construction of a semantic representation of words based on the statistical distribution of word co-occurrence in the text corpus; the meaning of words is represented by their corresponding vectors. We used model (i.e. Skip-gram Word2vec algorithm) presented by authors [63] to construct our semantic model that enclose the semantic representation of all words in the pre-processed Amharic Corpus. As we have reviewed, Skip-gram is a type of predict neural word embedding algorithm that uses the current word to predict the surrounding window of context words, and it has many advantages compared to earlier algorithms of distributional models. Next, we discuss briefly the model.

The Skip-Gram (SG) Model

The objective of the model is to find word representations that are useful for predicting the surrounding words in a given pre-processed Amharic texts. We'd like to predict c context words having one target word on the input as it is shown in *Figure 4-2* (Mikolov et al [63]). More formally, given a sequence of training words $w_1, w_2, w_3, \ldots, w_T$, the objective of the model is to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} \log \mathrm{p}\left(w_{t+j}|w_t\right) \qquad (4.2)$$

Where -c and c are limits of our context window (size of context window). The basic model formulation defines $p(w_{t+j}|w_t)$ using the softmax function:

$$p(wO|wI) = \frac{exp\left({v'_{wO}}^{T} v_{wI}\right)}{\sum_{w=1}^{W} exp\left({v'_{w}}^{T} v_{wI}\right)} \qquad (4.3)$$

where v_w and v'_w are the "input" and "output" vector representations of w, and W is the total number of words in pre-processed Amharic corpus.

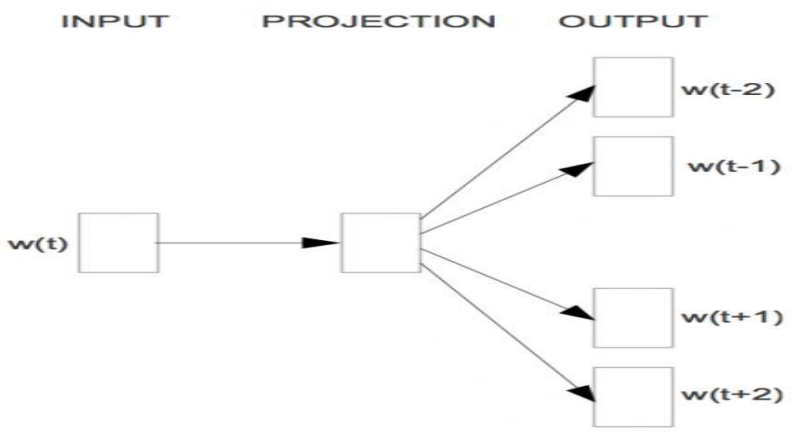

Figure 4-2: Skip gram model

4.3 Data Indexing

It is an offline process to index-linked datasets for the purpose of speeding up the query answering process. Particularly, it is important to match natural language expressions in the query with labels of vocabulary elements in the datasets. This component generates four independent indices corresponding to each element of the reference data model. This module has the following sub-modules;

4.3.1 Data Model Mapper

As we have stated above, our goal is to generate separate indices for resources (instances), properties, and classes by querying for the labels of such elements (or by separating natural language descriptors from URI) in the used datasets store. Thus, the main task of this sub-module is identifying data model categories for elements in the RDF(S) data model and then producing the normalized and stemmed form of them, and uses them as input for indexer. So, this section discusses how the proposed system categorizes each element of RDF statements into their data model categories (instances, properties, and classes). The task is composed of three major Activities as described below:

Activity 1: Defining Amharic Linked Data

As we have stated in Chapter two, we use RDF data model as a reference data model that represents information about resources with a <subject, predicate, object> structure (Triple structure), and Linked data as data integration. The subject denotes the instance, and the predicate denotes attributes of the resource and expresses a relationship between the subject and the object (i.e. < {instance ||class} - {property||type} - {instance

||class||value} >). Our linked dataset, which is important for both answering query and building indices, is collected, and interlinked from different domains (such as people, companies, films, music, locations, online communities, and scientific and government data) under Linked Data Principles (rather not a single dataset). The dataset is collected from a different source (such as Amharic DBpedia, Government open data portals, companies) having different data formats and then convert them to RDF data format using RDF free mapping tools.

Activity 2: Identifying Data Model Categories

To categorize elements of RDF to their data model, we consider two cases: the lexical categories of the label of elements and Link Types of RDF graph, and we implemented Algorithm 4-1. We discuss each case below:

Case 1: Link Types RDF Data Model

The following link types are identified:

Class link: Expresses a relationship between an instance and the unary predicate (i.e. < instance – type – class>). The link connects unary predicates with a 'is a' semantics. For example, *dbpedia: አድስ አበባ is a dbpedia: ከተማ.* In this type of link, the subject (*አድስ አበባ*) refers resource and the object (*ከተማ*) refers class, 'is a' refers type property.

Relation link: Expresses the relationship between two instances (i.e. < instance/class - property- instance/class>). In this type of link two different links are identified. The first link is where both subject and object are resources (i.e. < instance – property - instance>). For Example, dbpedia: አድስ አበባ dbpedia:ዋና ከተማ dbpedia: ኢትዮጵያ (i.e. Both አድስ አበባ and ኢትዮጵያ refers resources). The second link is where both subject and object are class (i.e. < class – property – class >). So, we need to identify lexical categories of the subject and the object. The predicate simple mapped to the property data model.

Attribute link: Expresses the attribute of an instance (i.e. < instance - property - value>). In this case the subject can be resources or class so we need to identify lexical categories of the subject. But the object is simple literal value so it is mapped to whether numeric values or instance (if values are strings). Here the attributes are mapped to the property data model.

Co-reference link: Expresses the identity or equivalence relationship between two instances, two properties or two classes (e.g. owl:sameAs, rdfs:equivalentClass, rdfs:equivalentProperty).

Description link: A link which provides a natural language descriptor for the dataset element (e.g. rdf:label, dcterms:description, rdfs:seeAlso).

Taxonomic link: terminology-level link which expresses a specialization/generalization relation (taxonomical) between classes or properties (e.g. rdfs:subClassOf, rdfs:subPropertyOf).

Case 2: Lexical Categories for the Data Model Elements

We used this case to distinguish whether the subject/object is instance or class in the above two links (i.e. Attribute link**,** Relation link). To get their Lexical category we used POS done by [84].

Instances: Map to named entities which refer to the description of entities. A named entity is defined by one or more proper nouns (NNP) in a noun phrase (NP).

Classes: Classes are unary predicates which map to non-rigid designators. Non-rigid designators are descriptors for sets of instances. A class is defined by one or more nouns (NN), adjectives (JJ), adverbs (RB), superlatives (JJS, RBS)) in a noun phrase (NP) or adjectival phrase (AP).

```
Input: ALD: Amharic Linked data
Intermediate: TEle: Triple Elements,
   PSe: POS tag of Subject,
   POb: POS tag of Object,
   LType=Link Type
Output: LI: List of Instances,
        //normalized and stemmed set of individuals
         LP: List of Properties,
         LC: List of class,
         LV: List of Value,

Begin:
   For each triple t in Amharic Linked data ALD
```

```
TEle = Tokenize (t, ' ')
 // returns ordered set of triple elements {s,p,o}
LType= identifyLinkTypes(TEle)
If LType= "Class Link" then
   LI.add(TEle[0])        LC.add(TEle[2])
els if LType= "Relation Link" then
     PSe =POS(TEle[0]))
     //part of speech of subject and object
     POb = POS(TEle[2]))
     If  PSe="NNP" and POb="NNP" then
      //both subject and object are entities
       LI.add(TEle[0])    LP.add(TEle[1])
       LI.add(TEle[2])
     els
        // both subject and object are descriptors
       LC.add(TEle[0])     LP.add(TEle[1])
       LC.add(TEle[2])
     end if
 els if LType= "Attribute Link" then
     PSe =POS(TEle[0]))
     POb = POS(TEle[2]))
     If  PSe="NNP" then  //subject is named entity
        LI.add(TEle[0])          LP.add(TEle[1])
        LV.add(TEle[2])
     els        // subject is descriptor
        LC.add(TEle[0])       LP.add(TEle[1])
        LV.add(TEle[2])
     end if
els If LType= "Co-reference Link" then
     PSe =POS(TEle[0]))
     POb = POS(TEle[2]))
      If  PSe="NNP" and POb="NNP" then
        LI.add(TEle[0])      LI.add(TEle[2])
      Els
```

```
            LC.add(TEle[0])      LC.add(TEle[2])
            end if
      els If LType= ”Description Link” then
            PSe =POS(TEle[0]))
            POb = POS(TEle[2]))
            If  PSe=”NNP” and POb=”NNP” then
               LI.add(TEle[0])     LI.add(TEle[2])
            Els
             LC.add(TEle[0])     LC.add(TEle[2])
            end if
      end if
End for
LI=normalize(LI)    LP= normalize(LP)
LC= normalize(LC)   LV= normalize(LV) //normalize each
set

LI=stemming(LI)    LP= stemming(LP)
LC= stemming(LC)   LV= stemming(LV)   // stemming each set
```

End

Algorithm 4-1: Data Model Mapper

Activity 3: Normalization

Preprocessed activities discussed in Section 4.2 are used to preprocess identified lexical categories list. Such as normalizes spelling variations by changing the different forms of a character into one common form and removal of extraneous characters and stemming are included.

4.3.2 Indexer

As we have stated above, the major goal of data indexing is creating indices for the RDF elements, this sub-component does the main task of indexing through requesting vector of target term and by calculating TFIDF of target term, it implement Algorithm 4.2. The task is composed of two major Activities as described below:

Activity 1: Requesting Terms Vectors

This activity is responsible to extend the formal semantics of RDF symbol with a distributional semantic description by requesting context vectors of terms from the trained distributional model. It begins by acquiring each target terms from each list of RDF elements (list of instances, list of classes, list of properties, list of values), and then request context vectors for each target terms from the pre-trained embedding model. If the target term is composed of more than one words (phrase), it asks for context vector for each word and calculates element-wise average sum. More formally, acquired target term T consists of words w_1 , w_2 , w_3 , ... , w_n with their corresponding context vectors (CV) $\overrightarrow{w_1}$, $\overrightarrow{w_2}$, $\overrightarrow{w_3}$ … , $\overrightarrow{w_n}$, the context vector of term ($\vec{T}$) given by:

$$\vec{T} = \sum_{j=1}^{dim} \frac{1}{n} \sum_{k=1}^{n} w_{k_j} \qquad (4.4)$$

Where dim is a dimension of context vector and **n** is the number of words in the target term.

Activity 2: Calculating weighting TFIDF of Terms

Its objective is to determine how a term is discriminative in relation to the relative distribution of other terms in the dataset. To do this we use a keyword-centric indexing approach, originating from the information retrieval research community [66]. Similarly to the way in which text documents are determined by a set of keyword terms in the classical information retrieval setting, each RDF documents is determined by a set of target terms. Our structure consists of inverted indices, where each row represents target terms and each column represents RDF documents.

More formally, Let S is the set of symbols (i.e. instances, properties, classes) in the preprocessed Amharic linked dataset and T is the set of target terms associated with the symbols S. Each $t \in T$ is composed of a set of k words. Let K be the set of all words in the dataset Lexicon. Let $w_{i,j} > 0$ be a weight associated with each word k_i contained in RDF document d_j , where for a k_i word not contained in a document d_j , $w_{i,j} = 0$. In this work we use TF/IDF weighting scheme for calculating weight associated with each term (The approach used in this study for term weighting is developed by modifying the

classical TF/IDF method used in lucene indexer) [67]. That means the TFIDF of target term is the average sum of each words TFIDFs' be present in the term. Where,

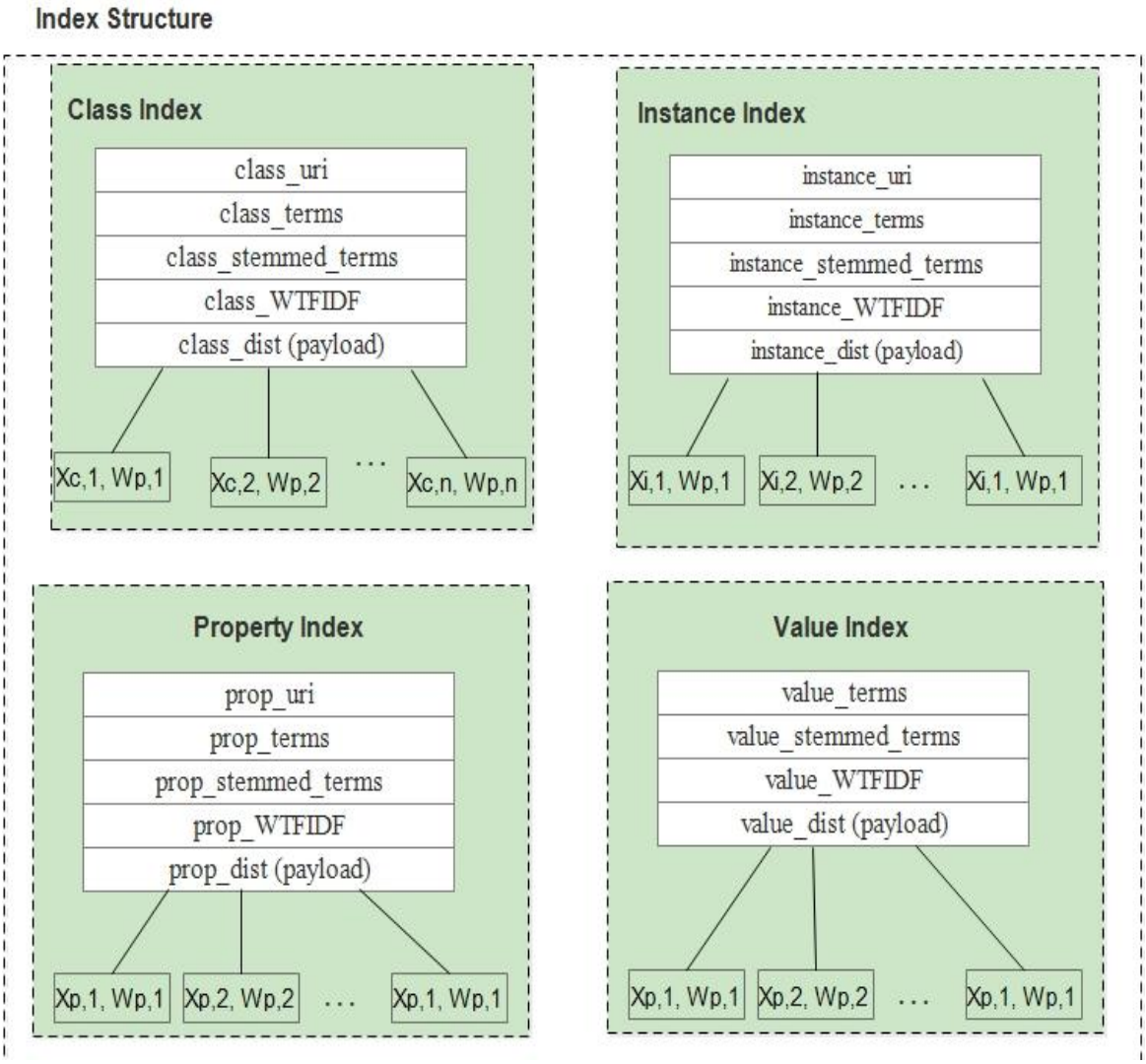

Figure 4-3: Lucene indices structures

Term Frequency (TF)

Let $freq_{i,j}$ be the frequency of term k_i in the RDF document d_j. Let count (d_j) is the number of words inside the document d_j. The normalized term frequency $tf_{i,j}$ is given by:.

$$tf_{i,j} = \frac{freq_{i,j}}{count(d_j)} \qquad (4.5)$$

Inverse Document Frequency (IDF)

Let n_{k_i} be the number of RDF documents containing the word k_i and N the total number of Documents in the dataset. The inverse document frequency idf_i for the word k_i is given by:

$$idf_i = log \frac{N}{n_{k_i}} \quad (4.6)$$

Term Frequency/Inverse Document Frequency

The final TF/IDF weight value of word based on the values of tf and idf is defined as:

$$w_{i,j} = tf_{i,j} \times log \frac{N}{n_{k_i}} \quad (4.7)$$

Thus, TFIDF of the term given by:

$$TfIdf_T = \frac{1}{n} \sum_{k=1}^{n} w_{tfidf_k} \quad (4.8)$$

Thus, the weight given by TF/IDF provides a measure on how a target term is discriminative in relation to the relative distribution of other terms in the dataset

```
Input:  LI: List of Instance terms and their urls,
        LP: List of Property terms and their urls,
        LC: List of class terms and their urls,
        LV: List of Value terms,
Intermediate: WTfIdf: weight TfIdf of term,
              Trmvector: weight term vector,
              EDim: Embedding Dimension,
              TfIdf = TfIdf of word
Output: InIndex: Instances index //index of instance terms
        PIndex: Properties index,
        CIndex: class index,
        VIndex: Value index
Begin:
   for each term t, url u in List of Instances LI
       Tstem=stemming(t)                //stemming term
       trm = Tokenize (Tstem, ' ')
       for each word w in trm
          WrdsVec=[][]  WTfIdf=0.0  indx=0
          EDim=embeddingDim()
          TfIdf=calculateTfIDF(w)  //calculating TfIfd of word
          WTfIdf= WTfIdf + TfIdf   // Averaging TfIdf of term
          indx++       //increment indx
```

```
    end for
    for n=1 to EDim  //calculate multi-words term vector
       Trmvector[n]=0.0
       for k=0 to size(wrdsVec)
         Trmvector [n]= 1/ size(wrdsVec) (Trmvector[n]
     +
         WrdsVec [k] [n]) //element-wise addition of vectors
       end for
    end for
    InIndex.add(u) InIndex.add(t)
InIndex.add(Tstem)
    InIndex.add(WTfIdf)      InIndex.add(Trmvector)
end for
for each term t, url u in List of Property LP
  //the same steps as the above steps to built Instance index
    PIndex.add(u) PIndex.add(t) PIndex.add(Tstem)
     PIndex.add(WTfIdf) PIndex.add(Trmvector)
end for
for each term t, url u in List of Class LC
   //the same steps as the above steps to built Instance index
    CIndex.add(u) CIndex.add(t)      CIndex.add(Tstem)
    CIndex.add(WTfIdf)      CIndex.add(Trmvector)
end for
for each term t in List of Value LV
   //the same steps as the above steps to built Instance index
   VIndex.add(t)           VIndex.add(Tstem)
   VIndex.add(WTfIdf)      VIndex.add(Trmvector)
end for
End
```

Algorithm 4-2: Data Indexing

Activity 3: Building Indices

It is responsible to build Lucene index by combining the final result of the above two activities. The core index structure consists of four indices, as depicts in Figure 4-3;

the instance index for instances, the class index for classes, the property index for properties and the value index for values. While uri stores the element URI, the field terms/stemmed terms cover the content of the parsed and stemmed URIs (word space), WTFIDF stores the weighted TFIDF of terms and the distributional context vector field, stores the distributional context vectors (distributional space). The distributional context vector is serialized as a byte array under the Lucene framework.

4.4 Query Template Generation

It is responsible to generate the corresponding domain independent SPARQL query template for user-provided natural language question input, and then the template is grounded to domain-specific entities via resource matching and disambiguation component. It gets natural language question from a user, process it, and produces a query template. To do so, first it preprocesses the given query using a shallow natural language processing technique, and then parsing semantically via deep neural learning and produces its corresponding semantic meaning representation, and lastly, semantic representations are finally translated into SPARQL templates. In general, it has the following three subcomponents:

4.4.1 Question Preprocessor

It is the former sub-module that admits user question and preprocess it via simple NLP tools such as normalizing and stemmer (words in question are normalized and stemmed). To do so, the preprocessor uses the same techniques to normalize and stemming as used in Corpora preprocessing (in Section 4.2.1). The output, the preprocessed question, is used by deep neural semantic parser sub-module as input to increase the performance of parsing instead of using the raw input question.

4.4.2 Deep Neural Semantic Parser

Semantic parsing (which captures the semantic structure of the user question) is a mapping of natural language utterances into a formal machine-understandable representation of its meaning (i.e. Logic form) [68]. Traditional ways of building a semantic parser [70-73] rely on high-quality lexicons, hand-crafted grammars and linguistic features which are limited by applied domain or representation. The rise of sequence to sequence (Seq2Seq) model [69] provides an alternative method to tackle the mapping problem and reduces the need for domain-specific assumptions, grammar

learning, and more generally extensive features engineering by taking into account semantic parsing as sequential problem.

Several types of supervision have been explored to train semantic parsing; full supervision, given a set of input sentences and their corresponding logical forms; training on utterance-denotation pairs and distant supervision to alleviate the annotation burden. In this work we propose to use a domain-independent neural semantic parsing that learns from natural language questions paired with meaning representations (full supervision based) and generate the semantic parser (model) to predict the meaning representation of new user's question. We use a recurrent neural network with long short-term memory (LSTM) model to encode sentences and decode logical forms. Our model can tolerate a number of additional difficulties for the Amharic language such rich morphology and larger freedom in sentence composition (e.g. የሀይለ ስላሴ ልጆች ስንት ናቸው? Or ሀይለ ስላሴ ስንት ልጆች አሏቸው? (i.e. how many children does Haile Sellase have?)). The following sub-section discusses specifications and components of our model.

Problem Setup

We briefly review the model presented by authors [74], which we base our model on. As we have stated, our model's training setup is fully supervised making use of utterance-logical form pairs. Thus, our aim is to learn a model which maps natural language input q =$x_1 \ldots x_{|q|}$ to a logical form representation of it meaning a =$y_1 \ldots y_{|a|}$. The entire model is trained end-to-end (depicted as Figure 4-4) by maximizing p (a\q):

$$p(a\backslash q) = \prod_{t=1}^{|a|} p(y_t | y < t, q) \tag{4.9}$$

Where y<t =$y_1 \ldots y_{t-1}$.

The model consists of an encoder which encodes natural language input q into a vector representation and a decoder which learns to generate $y_1, \ldots, y_{|a|}$ conditioned on the encoding vector.

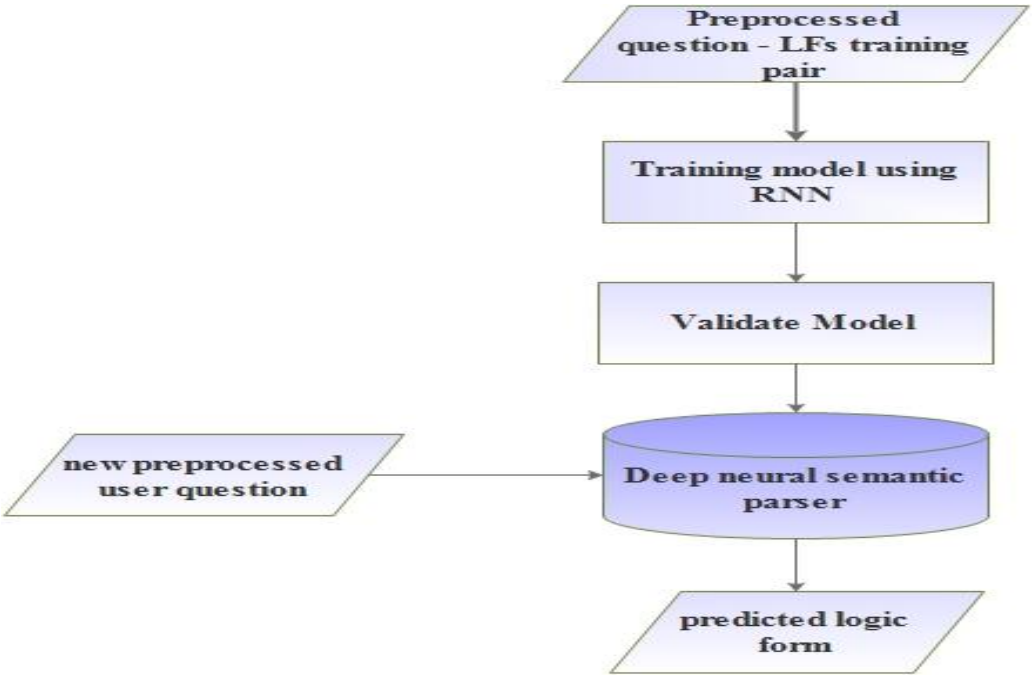

Figure 4-4: Deep neural semantic parser model learning

Sequence-to- Sequence Model (Sec2Sec)

This model considers both input "q" and output "a" as sequences. It is an encoder-decoder model (as shown in figure 4.4) with two different L-layer recurrent neural networks with long short-term memory (LSTM) neuron cell to handle long-range dependency (i.e. long user question).

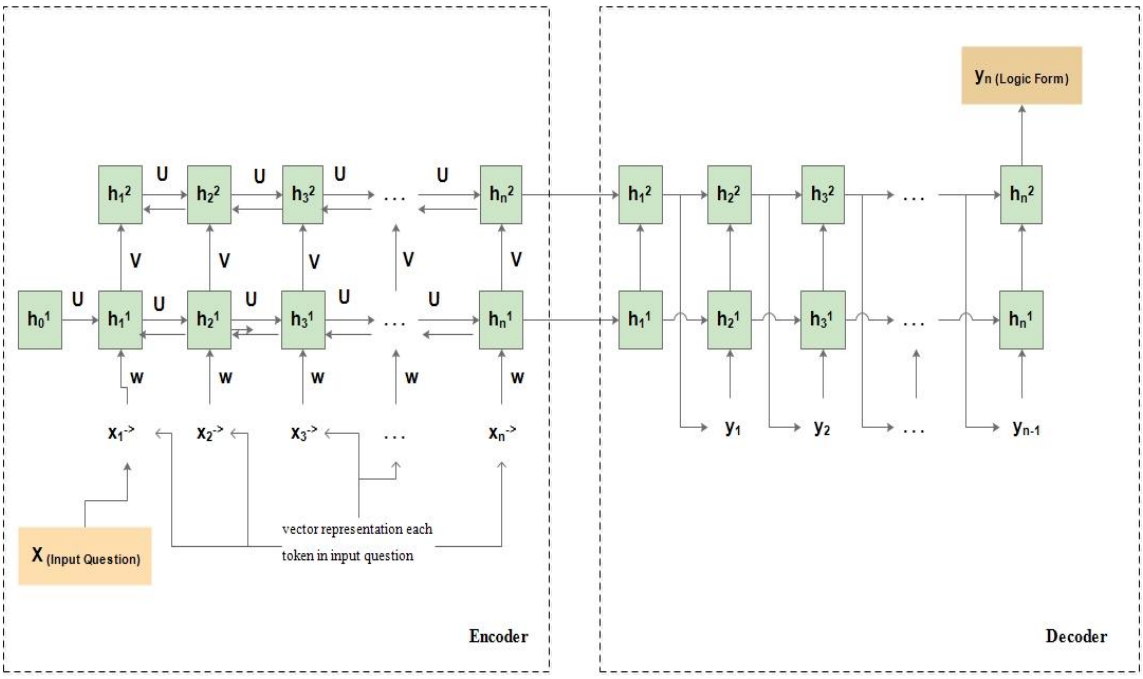

Figure 4-5: Our Sequence-to- Sequence RNN Model

We also extend the model with Bidirectional RNN to embrace the context of input and employing attention-based copying mechanism to learn soft alignment at each time step of the decoder. Below we briefly discuss the basic component of the model (encoder, decoder), and attention mechanism we used in this work.

Encoder: The encoder converts the input (user question) sequence $x_1 \dots x_{|q|}$ into a sequence of context sensitive embeddings $b_1 \dots b_{|q|}$using a bidirectional RNN. First, a word embedding function $\emptyset^{(in)}$maps each word x_i to a fixed-dimensional vector. These vectors are fed as input to two RNNs: a forward RNN and a backward RNN. The forward RNN starts with an initial hidden state h_0^F, and generates a sequence of hidden states $h_1^F, \dots, h_{|q|}^F$by repeatedly applying the recurrence:

$$\mathrm{h_i^F} = \mathbf{LSTM}\left(\emptyset^{(\mathrm{in})}(\mathrm{x_i}), \mathrm{h_{i-1}^F}\right) \qquad (4.10)$$

where LSTM refers to the LSTM function being used**.**

The backward RNN ($\mathrm{h_i^B}$) similarly generates hidden states $h_{|q|}^B, \dots, h_1^B$ by processing the input sequence in reverse order. Finally, for each input position i, we define the context-sensitive embedding b_i to be the concatenation of $\mathrm{h_i^F}$ and $\mathrm{h_i^B}$.

Decoder: our decoder is a sequence decoder that generates output tokens one at a time. Once the tokens of the input sequence $x_1 \dots x_{|q|}$ are encoded into vectors, they are used to initialize the hidden states of the first time step in the decoder. Next, the hidden vector of the top most LSTM h_t^L in the sequence, decoder is used to predict the t-th output token as:

$$p(y_t|y < t, q) = \mathbf{tanh}(W_0 h_t^L)^T e(y_t) \qquad (4.11)$$

Where W_0 is a parameter matrix, $e(y_t)$ a one-hot vector for computing y_t's probability from the predicted distribution, and tanh is Activation function which is adopted in the output layer to predict each word's probability.

Attention Mechanism: As shown in Equation (4.9), the hidden vectors of the input sequence are not directly used in the decoding process. However, it makes natural sense to integrate encoder-side information (in the form of a context vector) at each time step of the decoder to better predict the current token. Our attention model takes the alignment history into consideration (it helps to adjust future attention and guide the decoder towards unprocessed source words) through explicitly modeling the decoding

coverage of the source words. In order to find relevant encoder-side context for the current hidden state h_t^L of decoder, we compute its attention score with the k-th hidden state in the encoder:

$$s_k^t = \frac{\exp\{b_k.\ h_t^L\}}{\sum_j^{|q|} \exp\{b_j.\ h_t^L\}} \quad (4.12)$$

Where $b_1 \ldots b_{|q|}$ are the hidden vectors of top-layer of encoder. Then, the context vector is the weighted sum of the hidden vectors in the encoder:

$$c^t = \sum_{k=1}^{|q|} s_k^t\, b_k \quad (4.13)$$

Instead of Equation (4.9), we use this context vector which acts as a summary of the encoder to compute the probability of generating y_t in our model as:

$$h_t^{att} = \mathbf{tanh}(W_1 h_t^L + W_2 c^t) \quad (4.14)$$

$$p(y_t | y < t, q) = \mathbf{tanh}(W_o h_t^{att})^T e(y_t) \quad (4.15)$$

Where W_0, W_1, W_2 are the three parameter matrices and $e(y_t)$ a one-hot vector for computing y_t's probability from the predicted distribution.

1. a. ኢትዮጵያ ውስጥ ስንት ሀይቆች አሉ?
   b. answer (count (A) , ሀይቅ (A) , መገኛቦታ (A , ኢትዮጵያ))
   C. select count (?A) where {
   ?A rdf:type ?c.
   ?A ?p ?z
   }
   slots:
   - (?c, class, ሀይቅ)
   - (?p, property, መገኛቦታ)
   - (?z, resource, ኢትዮጵያ)

   Where, slots are missing elements of the query that are filled with appropriate URIs.

4.4.3 Template Generator

This subcomponent translates the semantic representation (logic form) of user question, resulting from the previous subcomponent, into domain independent SPARQL template. The template specifies the query type and the operation types as well as the number and

form of its triples. Another part of a template is the slots, which provide an appropriate URIs for the subject, predicate, and object of a triple. They are a tuple of a variable, the type of the intended URI (resource, class or property), and the natural language expression; (variable, intended URI, natural language expression). For example, for the question (1a) with its logic form (1b) the SPARQL template is given in (1c) is built (i.e. aggregation query type).

This component utilized 14 offline manually built up templates. Our template selection is based on the query type, its query form, and operation type as presented in Algorithm 3. We have identified three types of question and their corresponding query form (i.e. factoid or list – Select, Yes/No – Ask, Definition – Select), and four types of operation such as filter, aggregation, superlative, comparison.

```
Input:  LF: Logic form,
        STs: SPARQL Templates
Intermediate: AnsPart: Answer part of Logic Form,
              TriPart: Triple part of Logic Form
Output: SST: Selected SPARQL Template
Begin:
1. SepLF= Tokenize (LF, ",")
   AnsPart=SepLF[0]
   TriPart=SepLF[1-]
   If AnsPart contain "?" and TriPart  contain "ያንሳል"
      SST="Template 2" //Question type="yes/no" Operation type=" comparison"
   end if
   If AnsPart contain "?" and TriPart  contain "ይበልጣል"
      SST="Template 1"  //Question type="yes/no"      Operation type="
   comparison"
   end if
   If AnsPart contain "who" and TriPart  contain "ይበልጣል"
      SST="Template 4"  //Question type="select"      Operation type="
   comparison"
   end if
   If AnsPart contain "who" and TriPart  contain "ያንሳል"
      SST="Template 5"  //Question type="select"      Operation type="
   comparison"
   end if
   If AnsPart contain "?"
      SST="Template 3"      //Question type="yes/no"   Operation type="
   "
   end if
   If AnsPart contain "count"
      SST="Template 6" //Question type="select"      Operation type="
```

```
    aggregation"
    end if
    If AnsPart contain "count" and TriPart  contain "በላይ"
       SST="Template 7"    //Question type="select"      Operation type="
    aggregation+filter"
    end if
    If AnsPart contain "count" and TriPart  contain "በታች"
       SST="Template 8"   //Question type="select"      Operation type="
    aggregation+filter"
    end if
    If AnsPart contain "A" and TriPart  contain "argmax"
       SST="Template 9"   //Question type="select"      Operation
    type="superlative"
    end if
    If AnsPart contain "A" and TriPart  contain "argmin"
       SST="Template 10" //Question type="select"      Operation
    type="superlative"
    end if
    If AnsPart contain "A" and TriPart  contain "በላይ"
       SST="Template 11" //Question type="select"   Operation type="filter"
    end if
    If AnsPart contain "A" and TriPart  contain "በታች"
       SST="Template 12" //Question type="select"    Operation type="filter"
    end if
    If //none of the above conditions
       SST="Template 13" //Question type="select"    Operation type="entity
    description"
    end if
End
```

Algorithm 4-3: Template Generator

4.5 Resource Matching and Disambiguation

This component is responsible to ground domain independent SPARQL query, which has generated from the previous component to a given linked dataset to produce domains specific query. To do so, it first produces all candidate resources for each slot in the template, together with their URL and a matching score, by considering their type. Secondly, for each triple pattern (TP) in SPARQL template, data resources are sorted based on their coverage. Thus, the resources that have candidate matches for all or most of the terms in the TP are selected first. Then for each TP and each covering graph with candidate matches, the search for basic graph patterns (BGPs) is performed through an exploration of datasets. Finally merging of BGPs across resources is done by finding join

term, as well as one BGP with a common mapping. In general, it has the following sub-components;

4.5.1 Resource Matching

This component retrieves and assigns a score to resources (i.e. instances, classes, properties, and values) that are similar to the input string s from the given sub-indices for each data model category. The result is a Mapping Table per linguistic term, containing the matching score and the candidate URIs for each underlying dataset to be queried. The matching is based on exact (or approximate) string index searches, string distance metric and distributional semantic index searches over the entity labels for instances, properties, and concepts, as well as literal values. We used those matching techniques to rank and select only matches over a minimum threshold and up to a maximum number. This component can also return ranks of candidate terms that are pruned.

In this work, we performed two major techniques for candidate terms retrieval and ranking; term-based matching and distributional semantic matching. Instance search prioritizes term-based matching, due to the lower variability in the naming of instances and to the potential higher dimensionality of the distributional vector space (due to the proportionally higher number of instances in the linked datasets). But the other types of search (i.e. classes search, property search, and value search) used both matching techniques. The search operation consists of mapping the terms in the SPARQL template slots to the terms in indices.

In term-based searching, given a query term (i.e. term in slots) q^t in each data model categories over the term indices (i.e. corresponding to each data model index), the ranking function $\emptyset(q^t, t)$, $\forall t \in T$ is given by a combination:

- TF/IDF of terms in the indices: $s_{\frac{tf}{idf}}(q^t, t)$
- String similarity function: between the indices terms and the query term $sim_{dice}(q^t, t)$. Prioritizes closer string matching between query terms and indices terms. We used the Dice similarity coefficient [75] for comparing the similarity of two terms.
- Node cardinality (i.e. prominence score): number of triples $n(t)$associated with t.

Input: T: Set of Terms which are Indexed,
q^t: Query Term,
LLS: lucene Index space
$\eta_{tf/idf}, \eta_{dice}$: thresholds

Intermediate: $\emptyset_{tf/idf}$: tf/idf value of term,
$\emptyset_{dice}$: Dice Coefficient of term,
$\emptyset_n$: Number of triples associated to term

Output: R_t: set of matched and ranked terms to the query term q^t

Begin:

```
for each term t in Set of Terms T
  ∅_tf/idf = s_LLS(q^t, t)
  If ∅_tf/idf > η_tf/idf then
    ∅_dice = sim_dice(q^t, t)
    If ∅_dice > η_dice then
      ∅_n = n(t)
      R_t. add ( t → (∅_n, ∅_dice ))//add term with its two values (t → (∅_n, ∅_dice )) into set
    end if
  end if
end for
R_t= rank(R_t, ∅_n, ∅_dice) //rank terms based on their ∅_n, ∅_dice values
```

End

Algorithm 4-4: Term-based Search

$s_{tf/idf}(q^t, t)$ and $sim_{dice}(q^t, t)$ are used as filters: terms with values below a threshold are filtered. $sim_{dice}(q^t, t)$ and $n(t)$ are used as ranking functions. The term search returns a list of URIs associated with the term t which has at least one associated matching word to q^t . For all the terms containing at least one of the keywords associated with the query, the list is ranked according to string similarity. The ranking policy based on the node cardinality states that for homonymous terms, more popular

instances are prioritized. The ranking algorithm for the term search is given by Algorithm 4-4. This algorithm is similar for all type of search (i.e. instances search, properties search…).

Input: T: Set of Terms which are Indexed,
q^t: Query Term,
LLS: lucene Index space
η_{dist}: thresholds
DSM: Distributional semantic model

Intermediate: $\emptyset_{\text{dist}}$: distributional semantic relatedness value,

Output: R_t: set of matched and ranked terms to the query term q^t .

Begin:

$\overrightarrow{q^t}$ = weighted_word_embedding(DSM, q^t)// *Average vector of query term*
for each term t in Set of Terms T
$\emptyset_{dist}$ = $s_{LLS}\left(\overrightarrow{q^t},\vec{t}\right)$//*compute semantic relatedness between two vectors* ($\overrightarrow{q^t},\vec{t}$)
If $\emptyset_{dist}$ > η_{dist} then
R_t. add (t → $\emptyset_{dist}$)//*add term with its value* (t → ($\emptyset_{dist}$)) *into set*
end if
end for
R_t= rank(R_t,$\emptyset_{dice}$) // *rank terms based on their* $\emptyset_{dist}$ *values*

End

Algorithm 4-5: Distributional semantic search

The distributional semantic search operation is defined by the computation of the semantic relatedness measure between the corresponding distributional vector of the query term q^t and terms vector in term indices. We compute the cosine similarity [76]

between two distributional vectors to calculate the semantic relatedness measure between two terms. If query term is multi-words, we first request vector for each words and then we compute weight vectors for a term. The ranking algorithm for the distributional semantic search is given by Algorithm 4.5.

4.5.2 Disambiguation and Merging

As we have stated in Chapter two, our dataset which is essential to fill users need is an interlinked of different datasets employ heterogeneous schemas and each one may only contain a part of the answer for a certain question (i.e. Queries require fused information). Accordingly, we need to construct a federated formal query across different datasets. Thus, this component is in charge for disambiguating and ranking resource datasets, where each triple pattern (**TP**) in the SPARQL template will map, based on their coverage of candidate URLs, and merging of triples across datasets. Finally, it gives us candidate federated SPARQL templates. This component does the following two activities.

Activity 1: Disambiguating Datasets

This activity performs disambiguating of datasets and returns candidate BGPs for each TP in the given SPARQL Template. It has the following steps:

Step 1.1: For each TP, datasets are sorted based on their coverage. Thus, datasets that have candidate URL matches for all or most of terms in the TP are selected first. Here matching of candidate URLs and terms in TP is done by taking into account data model types of both to reduce search space.

Step 1.2: Then for each TP and each covering dataset with candidate URL matches, the search for BGPs is achieved through exploration of datasets. The search of BGPs begins by taking the given TP pattern, but, if it doesn't exist we consider the reverse pattern (i.e. so, the search result is either <?s , ?P , ?o> or <?o , ?p , ?s>). Here, we may get more than one BGPs for one TP because of the number of candidate datasets we got for one TP (it depends on the number of candidate URL matches we had for terms).

Activity 2: Merging across Datasets

Partial answers from BGPs across datasets that are translations to different TPs need to be combined to generate complete answers. For BGPs to be merged they need to have at least one variable (binding) in common, as well as one BGP with a common mapping

(the subject or object) that corresponds to the join term. However, when merging across datasets, the join term may not necessarily be represented by the same URI across datasets, even if they refer to the same real-world entity. It applies the following steps to merge BGPs across datasets:

Step 2.1: The first step is to find the join term between each pair of BGPs to be merged. To do this, we first identify TPs to be merged and join term for them; in our case, the join term and TPs to be merged are explicitly identified in SPARQL template: this means TPs that can be merged have at least one similar term (i.e. syntactically similar or similar variable). But we obtain this explicit information from SPARQL template that enclosed TPs, not BGPs. Then we generate a table that includes merged BGPs and their corresponding join terms of each merges, and candidate matches of each join terms obtained from both merged BGPs.

Step 2.2: the second step is on-the-fly term co-reference to retrieve the common bindings for the join terms across datasets, which will likely have different URIs even if they represent the same real-world entity. Thus, we perform two types of merging: syntactic and semantic. The semantic merging relies on the linkage across terms through equivalence relations. We have identified owl.sameAs, rdfs: equivalejntClass and rdfs: equivalentProperty equivalence relations types in our dataset. To find the linkage between the join variables we have added new BGPs formed from join terms and those identified equivalence relations. For example, lets we have two join terms (?s, ?r) determined from two merged BGPs across datasets, thus we link this two terms by constructing new BGP from join terms and equivalence relations; like { <?s ?rel ?r> union <?r ?rel ?s> FILTER (?rel1 == owl.sameAs || rdfs: equivalejntClass || rdfs: equivalentProperty) }. However, as equivalence links across entities are often sparse or may not be present, we also perform syntactic merging. An index search is added to the merged GP to find those instances from each join with similar labels.

Finally, this component generates candidate federated query templates with URL slots that have to be ranked by the coming component to fulfill user information needs.

4.6 Query Ranking and Execution

4.6.1 Query Ranking

After identifying BGPs and their combination (in the same and across datasets), we arrive at a range of possible SPARQL query templates. The task now is to rank these

queries and to pick one, which is then used to retrieve the answer to the input question. The score of query computed as the average of the score of BGPs. As we have stated above, one TP in a template can be translated to several BGPs, as a result, we do have numerous query templates.

Given a TP that is to be translated, we compute score for each possible BGP that can be the best translator: for each BGP average score of each individual mapping (candidate entities) is computed (A higher score is assigned to BGPs that combine more candidate URL matches), thus each BGP is ranked based on their score. The score is defined as;

$$score(BGP) = 1/n \sum_{i=1}^{n} \emptyset(q^t, t_i) \quad (4.16)$$

Where n is the number of terms in BGP, $\emptyset(q^t, t_i)$ is the score of each term which is computed based on string similarity and node cardinality (i.e. $\emptyset(q^t, t_i)$ = $\sigma\ (\emptyset_{dice}) + (\sigma - 1)\ \emptyset_n)$, where the vaue of $\sigma \in [0, 1]$ is decides on the impact of similarity and prominence on the score term). But, this calculation is not applicable for semantic matching

The final score of each query template is then defined as:

$$score(Q) = 1/m \sum_{j=1}^{m} score(BGP_j) \quad (4.75)$$

Where $score(BGP_j)$ is score of each BGP_j (j=1, 2, , , m). Hence, finally we have ranked list of query templates for question.

4.6.2 Query Execution

Once a ranked list of SPARQL queries is available, we need to decide which of those ranked queries are to be executed to catch the answer. If only the highest-ranking query would be returned, the problem arises that most of those queries actually do not return a result. The reason for this is that the query ranking method of ours only takes limited information into account for reasons of efficiency. It uses string similarity, prominence of entities. However, this does not guarantee that the combination of triple patterns in a query is meaningful and leads to a non-empty result. Therefore, our system is executed and tests queries before returning a result to the user, and finally, it retrieves top-ranked answer to the user.

Chapter 5 - Experiment and Evaluation

5.1 Overview

In this Chapter, the set of experiments conducted to validate the proposed Semantic Question Answering over Amharic linked data approach are presented. We have followed some set of procedures to conduct the experiment. The test environment, the set of activities defined under the procedures, the experiment settings and the findings from the experiments are described in detail in the following sub-sections.

5.2 Experimental Procedure

In this sub-section, we describe datasets important for conducting different experiments corresponding to each component of the system and the overall system, and we present our experiment setting including programming languages.

5.2.1 Data Set Collection

Three different sets of data were required in order to undertake the experiments. The first set of data is a collection of Amharic structured data referred to as ***"****Amharic linked dataset****"*****.*** This data was used to build indices and to find an answer for user question. The source for the data was the Amharic version of DBpedia dump version 2016-10 which is downloaded online as turtle (ttl) serialization, and the other source was open data Web portals. Data obtained from the data portal was in .CSV and in .XML format where we converted them into RDF data format through RDB2RDF, mapping tool downloaded from the Internet. The collection was domains independent and encompasses interlinked heterogeneous sources via linked data principles explained in Chapter two. The dataset contains 30,000 RDF triples.

The second set of data is a collection of Amharic texts documents referred to as **"***Amharic reference corpora***"** which was gathered from different sources including Amharic news from WIC information center, Amharic Wikipedia dump and Amharic version of the bible. This data was used to build our distributional semantic model (word2vec). A total of 35877 Amharic text files (593.5 MB) has been collected, and then we preprocessed each file and minimize the size of corpus to 506.66 MB.

The third set of data is a collection of Amharic questions and their annotations referred to as ***"Amharic Natural Language questions".*** To the best of our knowledge, no benchmark for Amharic semantic query answering over Linked Data has been created so far. Thus, we created a benchmark consisting of 2300 queries on Amharic interlinked dataset for the purposes of our experimentation. This data was used to build and test deep semantic parser, and to test the overall system performance. The dataset is split into a training set of 2250 queries and 50 test queries to build semantic parser models. The training queries are annotated with corresponding Logic Form (LF) whereas the test set prepared into two formats: the first format enclose queries annotated with corresponding Logic Form (LF) in order to evaluate the built semantic parser model and the other format enclose queries annotated with corresponding SPARQL queries and answers to evaluate our system performance. The second format annotations are presented in XML format as shown in *Figure 5-1*.

```
<question id="10" answertype="resource" aggregation="false" order=" true" comparison=" false" >
  <string> ኢትዮጵያ የሚገኘው ረዝሙ ተራራ ማነው?</string>
  <query>
    SELECT ?uri WHERE {
      ?url rdf:type <http://am.geo.org/resource/ተራራ>
      ?url  <http://am.geo.org/property/ክልል> ?o
       ?o<http://am.geo.org/property/ሀገር> <http://am.dbpedia.org/resource/ኢትዮጵያ>
      ?url <http://am.dbpedia.org/property/ከፍታ> ?B
    }
    ORDER BY DESC (?B) LIMIT 1 OFFSET 0
 </query>
  <answers>
    <answer>
     <uri> http://am.geo.org/resource/ራስዳሽን_ተራራ </uri>
     <string>ራስ ዳሽን </string>
    </answer>
  </answers>
</question>
```

In this work, we adopted FunQL (i.e. less expressive than lambda calculus but it is simple) as the semantic formalism of logic form. We used similar primitive domain-general functional operators defined by authors [77] to define FunQL formalism as shown in *Table 5-1*. The test data consists of 8 aggregate questions, 6 comparison questions, 8 yes/no question, 8 superlative questions, 5 questions require fused information across different dataset sources, 15 list, and factoid questions. Some of these questions with their corresponding annotations are presented on Annex F. We walk through the following steps to create the test data benchmark: first, we gathered queries from graduate classmates, and then secondly we annotated them via their corresponding LF and/or SPARQL query, and finally we prepared expert review questionnaires to validate the benchmark and evaluated by three independent language and semantic Web experts.

Table 5-1: Domain general functional operators to define FunQL formalisms

Operators	Description	Examples
Entity	creates a unary logical form whose denotation is a singleton set containing that entity	ቀዳማዊ ኃይለሥላሴ
Relation	Acts on an entity and returns as detonation the set of entities that satisfy the relation.	Logic Form ልጆች (ኃይለ ሥላሴ, A) corresponds to the question "የ ኃይለ ሥላሴ ልጆች እነማ ናቸው? "
Count	Returns the cardinality of a set of entities.	count(A, ልጆች (ኃይለ ሥላሴ, A)) corresponds to the question "ኃይለ ሥላሴ ስንት ልጆች አሏቸው?"
argmax or ***argmin***	operator returns a subset of entities whose specific property is maximum or minimum	Expression argmax(ልጆች (ኃይለ ሥላሴ, A), እድሜ) corresponds to the question "የኃይለ ሥላሴ የመጀመሪያ ልጅ ማነው? "
Filter	Returns a subset of entities where a comparative constrain is satisfied	Expression filter(ልጆች (ኃይለ ሥላሴ), >(እድሜ, 52)) corresponds to the question "ከ ኃይለ ሥላሴ ልጆች እድሜው ከ52 የሚበልጠው ማነው?"

The verification criteria on the questionnaire are used to ensure that the selected test data collections satisfy the characteristics necessary to support the evaluation of the thesis' hypothesis and core requirements (mainly, comprehensiveness of question set; the expressiveness of questions over open domain) and they are adopted from paper presented by authors [56]. Based on the experts' feedback, some questions have incorrect corresponding SPARQL queries and logic forms and they suggested that they have to be corrected. In addition, some experts suggested that the attribute fused is false in some question, this means it is not contributing to the diversification of the questions. Except for these comments and recommendations, the finding of experts result shows that the test collections are satisfied the core research requirements. Expert questioner form is attached as Annex E.

5.2.2 Experiment Setting

In this section, we discuss experiment settings including tools and programming languages to employ different experimentations corresponding to components of the system. In the next sub-sections, we describe each experiment.

Experiment Setting to Train Word Embedding Model

This experiment focused on training of semantic word embedding model. Here we performed three experiment settings based on the dataset we used to train the model, and then we built three different semantic models. The first setting utilized *"normalized but not stemmed Amharic reference corpora*" dataset, the second is *"normalized and stemmed Amharic reference corpora"* whereas the other setting utilized *"stemmed and sub-sampled Amharic reference corpora"* dataset to recognize the effect of data stemming and corpora sub-sampling to the rest of components and on the whole system. Other than dataset settings we specified hyper parameters (or training parameters) to train the models as shown in *Table 5-2*.

Table 5-2: model training parameters

Model	Dimensionality	Window size	Sub-sample	Negative	Freq.thresh
SG	300	10	$1 \times e^{-3}$	10	10

As we can see from *Table 5-2*, we used Skip Gram (SG) word2vec model with vector dimension of size 300, context window of size 10, and negative sampling with k values

of 10 as a maximization algorithm. We used the same training parameters for the above two mentioned dataset settings to build two different word embedding models. Both models were implemented in Python programming language with genism integration.

Experiment Setting to Train Deep Neural Parser

The model was trained by a prototype (one parts of the prototype) implemented in a python programming language with Theano integration [78]. Similar to the above setting, we prepared three settings; the first setting is based on *"normalized but not stemmed Amharic Natural Language questions"*, the second is based on *"normalized and stemmed Amharic Natural Language questions"* (i.e. Words in natural language questions were stemmed using HornMorpho) and lastly our setting is "*stemmed but without an attention mechanism*" . In all settings, we replaced word vectors for words that occur only once in the training set with a universal <unk> (unknown) word vector in the questions, but we but kept all tokens in the logical forms. To train our models we used deep LSTMs with 2 layers, with 200 cells at each layer and 200-dimensional word embeddings. Word embeddings were initialized with Word2Vec embeddings. The complete training details are given below:

- We initialized all of the LSTM's parameters with the uniform distribution between -0.08 and 0.08
- We used the RMSProp algorithm (with batch size set to 20) to update the parameters. The smoothing constant of RMSProp was 0.95.
- Gradients were clipped at 5 to alleviate the exploding gradient problem.
- The dropout rate was set to 0.5, which computes the softmax activation of the next action or token.
- We trained the model for 30 epochs with an initial learning rate of 0.1, and halved the learning rate every 5 epochs, starting from epoch 15.
- We used beam search with beam size 5 to generate logical forms during inference.

Java Programming language, the Lucene API (Lucene 3.7), and a number of other third-party Java libraries such as Jena framework (i.e. used to execute SPARQL query on the linked data) were used in implementing other components of our system/prototype. All experiments were implemented in a PC laptop with Intel(R) Core (TM) i3-5005U CPU 2GB Hz, 4GB memory RAM, and 500 GB disk running Windows' 8.1.

5.3 Evaluation

5.3.1 Offline Evaluation

Evaluation of Deep Neural Parser

The evaluation is based on the created test data benchmark on Amharic linked dataset. It comprises set of 50 questions over Amharic linked dataset, annotated with logic form. Each model is evaluated by calculating the bilingual evaluation understudy score (BLEU [79], is a metric for evaluating a generated logic form to a reference logic form) scores to get a quantitative idea of how well the model has performed. We computed different cumulative BLEU scores by adjusting weights; BLEU-1(1, 0, 0, 0), BLEU-2(0.5, 0.5, 0, 0), BLEU-3(0.33, 0.33, 0.33, 0) and BLEU-4 (0.25, 0.25, 0.25, 0.25) and results of models are presented in Table 5-3.

Table 5-3: Evaluation results of different Neural Semantic Parsers by calculating four cumulative BLEU scores

Scores / Models	BLEU-1**(1-gram)**	BLEU-2**(2-gram)**	BLEU-3**(3-gram)**	BLEU-4**(4-gram)**
NSPM with Stemmed	0.60	0.57	0.57	0.51
NSPM not Stemmed	0.60	0.55	0.55	0.49
NSPM Stemmed No Attention	0.50	0.48	0.47	0.44

As shown in Table 5.3, the NSPM (Neural Semantic Parser Model) with both Stemmed and attention mechanism is superior to other models in all BLEU scores. It achieved 0.51 score of BLEU-4 (i.e. which provides an upper bound on what we might expect from this model), and outperforms NSPM Not Stemmed and NSPM with Stemmed and No attention mechanism by 2.47% and 7.41% respectively (taking BLEU-4 score). This is to be expected since the hidden vectors of the input sequence were directly used in the decoding process (i.e. means it considers relevant information from the encoder-side to better predict the current token), and different forms of each token appeared in the training questions were changed into their stemmed form. All individual BLEU scores for each test question of NSPM with Stemmed model is presented on Annex C.

Comparing our reference model (i.e. NSPM with Stemmed) to other deep neural based semantic parser model developed for other languages, it performs less score. This is due

to: *the stemmer* we used has less quality; For instance, let's take these two questions which they are taken from stemmed question set "ሀዋሳ ከተማ የሚለማመዱበት ስታድየ ተመልካች የመያዝ አቅም ስንት ነው" and "አበበ ቢቂላ ስታድየ ተመልካች በመያዝ አቅም አዳማ ስታድየ ይበልጣል", words የመያዝ (from the first question) and በመያዝ (from the first question) stemmed into different forms(but they should have the same stemmed form, መያዝ), and they are expected to be mapped to the same token (ተመልካች_አቅም) in corresponding logic forms. Thus, the probability of each word mapped to token ተመልካች_አቅም is smaller than the probability of their stemmed form (i.e. መያዝ). In general, stemmer quality has a direct impact on the performance of our reference semantic parser.

The second case is *rare words***;** because the training question set size is relatively too small, some question words and semantically light expressions, such as the verbs to be and to have, and prepositions are rare in the training set, which makes it hard to estimate reliable parameters for them (or difficult to handle their context and meaning). Our data size has also affect to get good precision for each question forms of the same question, and question types (simple-complex). The last case is a *number of layers*; we used two layers for encoder and decoder models, where they don't provide more representational capacity for the model. This is due to the performance bound of a computer we used for experimentation is limited (need GPU) to train the model with >2 hidden layer and more epochs (we leave this future work).

Evaluation of Word Embedding Models

We evaluated word embedding models described above based on collected term pairs set (wordsim100, provide human annotated scores of relatedness between term pairs). We used word relatedness evaluator (i.e. one type of intrinsic evaluator [80]) to measure how well human perceived relatedness is captured by the word vector representation; we measured the correlation between the mean score provided by the 10 language experts datasets and cosine scores calculated by our models, using Spearman's correlation coefficient [81]. The questioner form used to collect relatedness scores to pairs of words from experts, and the collected human's mean score and cosine scores obtained from embedding models are attached as Annex A and Annex B respectively. We computed the Spearman's correlation coefficients between each embedding model and human-provided mean score using statistical software tool, IBM SPSS statistics (version 19). Table 5-4, Table 5-5, and Table 5-6 show the result of each computation.

Table 5-4: It presents Spearman's correlation coefficient rs, is 0.64, it's statistically significance value (ρ=0.0) and the sample size (100) that the calculation was based on in a matrix, between WVM with Stemmed cosine score and human mean semantic relation score

Correlations				
			WVM with Stemmed	Human mean Semantic _relation _score
Spearman's rho	WVM with Stemmed	Correlation Coefficient	1.000	.639**
		Sig. (2-tailed)	.	.000
		N	100	100
	Human mean Semantic _relation _score	Correlation Coefficient	.639**	1.000
		Sig. (2-tailed)	.000	.
		N	100	100
**. Correlation is significant at the 0.01 level (2-tailed).				

Table 5-5: It presents Spearman's correlation coefficient rs, is 0.56, it's statistically significance value (ρ=0.0) and the sample size (100) that the calculation was based on in a matrix, between WVM not Sampled cosine score and human mean semantic relation score

Correlations				
			WVM not Sampled	Human mean Semantic _relation _score
Spearman's rho	WVM not Sampled	Correlation Coefficient	1.000	.561**
		Sig. (2-tailed)	.	.000
		N	100	100
	Human mean Semantic _relation _score	Correlation Coefficient	.561**	1.000
		Sig. (2-tailed)	.000	.
		N	100	100
**. Correlation is significant at the 0.01 level (2-tailed).				

Table 5-6: It presents Spearman's correlation coefficient rs, is 0.43, it's statistically significance value (ρ=0.0) and the sample size (100) that the calculation was based on in a matrix, between WVM not Stemmed cosine score and human mean semantic relation score

Correlations				
			WVM not Stemmed	Human mean Semantic _relation _score
Spearman's rho	WVM not Stemmed	Correlation Coefficient	1.000	.430**
		Sig. (2-tailed)	.	.000
		N	100	100
	Human mean Semantic _relation _score	Correlation Coefficient	.430**	1.000
		Sig. (2-tailed)	.000	.
		N	100	100
**. Correlation is significant at the 0.01 level (2-tailed).				

Table 5-7: Spearman's rank-order correlation, rs, between the word relatedness scores of the word vector models and mean score provided by humans

	WVM with Stemmed	WVM not Stemmed	WVM not Sub-sampled
r_s	0.64	0.43	0.56

As shown in *Table 5-7*, WVM (Word Vector Model) with Stemmed is superior to other models in Spearman's correlation score. It achieved 0.64 scores of r_s. Our models and human's mean score have a positive correlation in relatedness assessment, with most of the correlation coefficient higher than 0.5, which means the relationship represented in vector space is consistent with human annotations. As a result, we suggested that developing word embedding via stemmed Amharic corpus is more effective to compute words semantic relatedness, and we used WVM with Stemmed as reference word embedding model.

5.3.2 Online Evaluation

In this evaluation, we evaluated the overall system performance under the created test data benchmark on Amharic linked data. We used relevance metrics [82] to measure the relevance of the results returned by the system and to test its effectiveness. We evaluated the system with the three different setting of semantic parser models and used WVM with Stemmed as a semantic model for all settings. Each question (and the averages for the whole query set) is evaluated with respect to precision (p@3), recall (r@3), and f-measure (f@3) (i.e. we consider all answers until ranked position 3) defined as follows:

Precision: provides a measure of how accurate is the answer set, i.e. the fraction of retrieved results that are relevant for the query, and it is given by the following expression:

$$p = \frac{\text{number of correct answers returned by sthe ystem}}{\text{number of all answers returned by the system}} \tag{5.1}$$

The average precision is given by:

$$\text{avg.}\, p = \frac{1}{\text{N}} \sum_{\text{i=1}}^{\text{n}} p_{\text{i}} \tag{5.2}$$

Recall: provides a measure of the completeness of the query set and consists of the fraction of relevant results that are retrieved, and it is given by the following expression:

$$r = \frac{\text{number of correct answers returned by system}}{\text{number of answers in gold standard answers}} \tag{5.3}$$

The average recall is given by:

$$\text{avg.}\, r = \frac{1}{\text{N}} \sum_{\text{i=1}}^{\text{n}} r_{\text{i}} \tag{5.4}$$

F-measure: is a harmonic mean which aggregates precision and recall into a single measure:

$$Fmeasure = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{5.5}$$

Table 5-8: Aggregate relevance results for the query results on each neural semantic parsing model

Measure Type	NSPM with Stemmed	NSPM not Stemmed	NSPM Stemmed No Attention
Avg. Recall	0.58	0.55	0.49
Avg. Precision	0.43	0.40	0.35
Avg. F-Measure	0.50	0.46	0.41

As shown in *Table* 5-8, Question answering system with NSPM with Stemmed is superior to the systems with other models in all relevance metrics. It achieved mean avg. F-Measure of 0.50 (50%), Avg. Precision of 0.43, Avg. Recall of 0.58. The medium-recall of the system showed us the effectiveness of the ability to survive with the conceptual/vocabulary (semantic gaps) and structural gaps, even where the performance of semantic parser is low. All relevance metrics for each test question of NSPM with Stemmed is presented on Annex D.

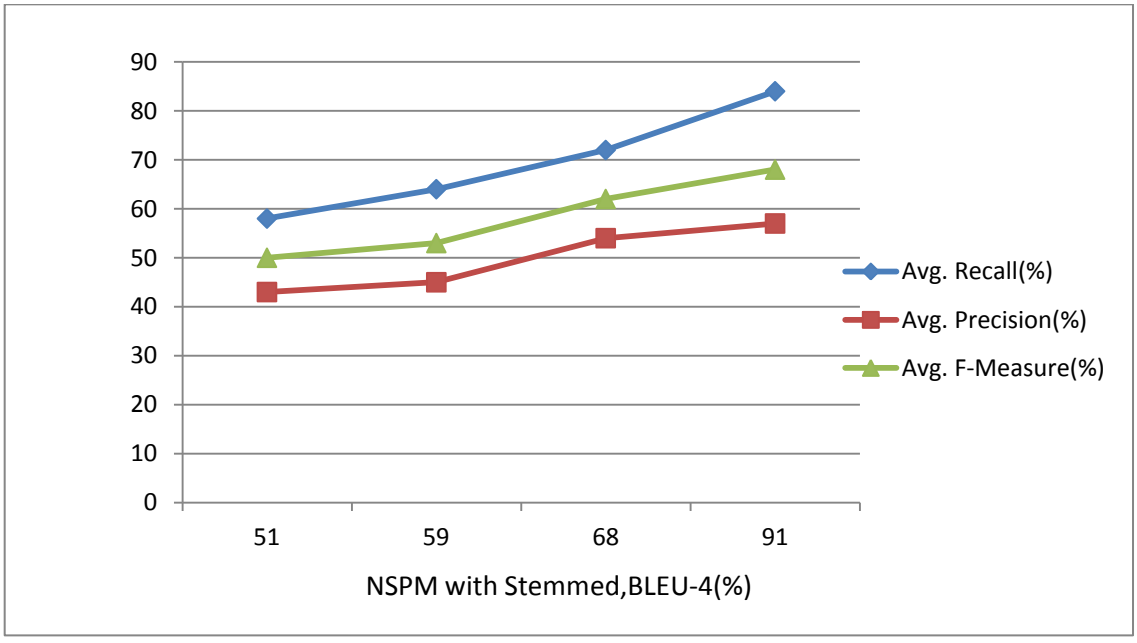

Figure 5-2: system performance curves for all relevance metrics on the NSPM with Stemmed (BLEU-4 score)

We also demonstrated the performance of our system by enhancing the performance of semantic parser. As shown in Figure 5-2, we have carried out four different experiments by raising the value of NSPM with Stemmed, and the result showed us, system performance is increased with parsing performance (NSPM with Stemmed) is increase,

indicates our system can capture faithfully the semantic structure of the natural languages in test set, thus complex questions containing quantifiers, comparatives, superlatives, and requiring fusion of information from different resources pose no problem, and the system can cope with the conceptual/vocabulary (semantic gaps) and structural gaps. This indicates if problems of our semantic parser we have discussed on section 5.3.1 will minimized for the future, our system will perform well.

Chapter 6 - Conclusion and Future Work

6.1 Conclusion

Since digital technology has emerged, a massive amount of data has been produced in a structured format. Large data is available from online transactions, social networking sites, statistical data, online data portals, and etc. In addition, data which had been held in paper or hard copy for a long time has been converted to soft copy to share them with the public. For instance, the Ethiopian government as part of the eOffice project to decide to change part of the paper-based reports to be converted into its equivalent electronic version. Accordingly, the Ethiopian government developed an open data Web portal that allows organization to publish their data. As these large collections of data contain a set of facts in multiple domains, mechanisms to support users interacting, querying and exploring data without the needs to understand its specific representation, lexicon and structure become a fundamental demand for contemporary data management.

We presented an approach to Amharic semantic suestion answering over linked data that relies on a deep linguistic analysis (by neural semantic parser) yielding a SPARQL template with slots that need to be filled with URIs. In order to fill those slots, possible datasets were disambiguated, and entities were identified using string similarity as well as semantic relatedness using distributional semantic model developed from text documents. The remaining query candidates were then ranked on the basis of scores attached to the entities and BGPs and finally top ranked query was selected to be executed as the final result.

One of the strengths of this approach is that the generated SPARQL templates capture the semantic structure of the Amharic question input. Therefore, questions containing quantifiers, comparatives, and superlatives do not pose a problem. In some cases, the semantic structure of the question and the triple structure of the query do not coincide; to solve this problem we investigated the relaxation of templates, such that the triple structure is not completely fixed but is discovered through exploration of the RDF data (i.e. this is the second strength of the approach). The other strength is that the system can answer questions requiring the fusion of information from different resources via disambiguating and ranking resource datasets and constructing a federated query across different datasets.

The overall system performance was evaluated under the created test data benchmark on Amharic linked data. We used relevance metrics to measure the relevance of the results returned by the system and to test its effectiveness. Each question (and the averages for the whole query set) is evaluated with respect to precision, recall, f-measure. We evaluated the system with the three different setting of semantic parser models. Question answering system with NSPM with Stemmed is superior to the system with other models in all relevance metrics. It achieved mean avg. F-Measure of 0.50 (50%), Avg. Precision of 0.43, Avg. Recall of 0.58. We also demonstrated the performance of our system by enhancing the performance of semantic parser. The result showed us increasing system performance with increasing parsing performance is increased.

6.2 Contribution

The contributions of this thesis work are summarized as follows:

- Making use of link type of RDF graphs and lexical categories (we used POS to determine lexical category) of the label of elements to categorize elements of RDF to their data model (Instances, Properties, class, and values).
- Extending the formal semantics of RDF symbol with a distributional semantic description by requesting context vectors of terms from the trained distributional model.
- Making use of deep learning approach to maps Amharic natural language question into its corresponding logical form representation.
- Making use of 14s offline built up query templates to translate logical form representation of user question into corresponding domain independent SPARQL template.
- Appling tfidf of terms, dice similarity coefficient for comparing the similarity of two terms, and node cardinality (i.e. prominence score) for filtering and ranking in term-based searching.
- Appling distributional semantic relatedness matching approach so as to determine the semantic relatedness between the query and dataset terms.
- Appling disambiguating datasets and merging across datasets to construct a federated SPARQL query for question requiring fusing of information across different datasets. Partial answers from different datasets to be merged they need to have at least one variable (binding) in common, as well as one BGP with a

common mapping (the subject or object) that corresponds to the join term. To do this, we perform semantic and syntactic merging. The semantic merging relies on the linkage across terms through equivalence relations. We have identified owl.sameAs, rdfs: equivalentClass and rdfs: equivalentProperty equivalence relations types. Syntactic merging is through an index search to find instances from each join with similar labels.

- Making use of graph exploration of the RDF data to coincide the semantic structure of the question and the triple structure of the query via the relaxation of templates.
- Preparing test data benchmark to evaluate the proposed model. The benchmark contains 50 questions and each question presented in XML format and has question ID, question type, answer type, corresponding logic form representation, corresponding SPARQL query, and list of one or more answers The test data consists of 8 aggregate questions, 6 comparison questions, 8 yes/no questions, 8 superlative questions, 5 questions require fused information across different dataset sources, 15 list and factoid questions.
- Appling word relatedness evaluator to evaluate word embedding models. Here we collected 100-word pairs (*wordsim100*), then provided to 10 language experts and finally we measure the correlation between the mean score provided by experts' datasets and cosine scores calculated by our models using Spearmans' correlation coefficient.

6.3 Future Work

In this research, we have made an attempt to explore the use of a new approach to build a semantic question answering. The following directions are pointed out so that this research can be further pursued.

- As it is mentioned in Section 4.4.2, the semantic parser is trained with full supervision, given a set of input sentences and their corresponding logical forms, but it is highly costly to annotate logical forms. Therefore, we will investigate ways to refine training supervision to alleviate the annotation burden.
- As described in Section 4.4.2.2, the number of layers we used to train encoder and decoder models is too small (2 layers), where they don't provide more

representational capacity for the parser. We recommend developing a semantic parser that is trained with >2 hidden layer (especially 4-layers) and more epochs.

- Provide an inference engine, which infers additional triples given a set of triples. For instance, when we look at question No 10 in the test data "ኢትዮጵያ የሚገኘው ረዝሙ ተራራ ማነው?" and it's corresponding logic form "answer (A, argmax ((ተራራ (A), መገኛ_ሀገር (A, ኢትዮጵያ), ከፍታ (A, B)), B))", the direct triple ?uri <http://am.geo.org/property/ሀገር> <http://am.dbpedia.org/resource/ኢትዮጵያ> cannot retrieve resources type of <http://am.geo.org/resource/ተራራ> in the given dataset. Therefore, the system should infer additional triples such as ?uri <http://am.geo.org/property/ክልል> ?o and ?o <http://am.geo.org/property/ሀገር> <http://am.dbpedia.org/resource/ኢትዮጵያ>.
- A lot of information is still available only in textual form, both on the Web and in the form of labels and abstracts in linked data sources. Therefore, we recommend developing a hybrid question answering that can retrieve answers for questions that required the integration of data both from linked data and from textual sources, processing both structured and unstructured information.
- Provide bilingual question answering, structured data on the Web (including DBpedia) and local organizations (open data portals) available in both Amharic and English languages form. Thus, it needs to integrate data from sources existing in both languages and allows natural language questions in both languages.

References

[1] “Ethiopian Government Open Data Portal [online],” https://www.data.gov.et/, Last accessed on Jan 20, 2018.

[2] Jens Lehmann a, Robert Isele g, Max Jakob e, Anja Jentzsch d, Dimitris Kontokostas a, Pablo N. Mendes f , Sebastian Hellmann a, Mohamed Morsey a, Patrick van Kleef c, S¨oren Auer a;h, Christian Bizer b, “DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia,” 1570-0844/12/$27.50 2012 – IOS Press and the authors

[3] DBpedia Dataset,http://wiki.dbpedia.org/services resources/datasets/dbpedia-datasets [accessed on 2018].

[4] James Baile, Franc Ois Bry, Tim Furche2, Sebastian Schaffert, “Semantic Web Query Languages,” Comp. by: RPrabhuGalleys0000861669 Date:22/9/08 Time:23:39:29

[5] Fekade Getahun, Genet Asefa, “Towards Amharic Semantic Search Engine,” MEDES '15, October 25-29, 2015, Caraguatatuba, Brazil © 2015 ACM. ISBN 978-1-4503-3480-8/15/10…$15.00

[6] Tessema Mindaye Mengistu and Solomon Atnafu, "Design and Implementation of Amharic Search Engine," in International IEEE Conference on Signal-Image Technologies and Internet-Based System - SITIS, 2009, pp. 318 - 325.

[7] Hassen Redwan, Tessema Mindaye, and Solomon Atnafu, "Enhanced Design of Amharic Search Engine (An Amharic Search Engine with Alias and Multi-character Set Support)," in AFRICON '09, Addis Ababa, 2009, pp. 1-6.

[8] Tewodros Hailemeskel Gebermariam, "Amharic Text Retrieval: An Experiment Using Latent Semantic Indexing (LSI) with Singular Value Decomposition (SVD)," Master’s Thesis, School of Information Studies for Africa, Addis Ababa University, Addis Ababa, Ethiopia, Unpublished 2003.

[9] Seid Muhie Yimam, Mulugeta Libsie “Teteyeq: Amharic Question Answering System for Factoid Questions,” Master’s Thesis, Computer Science Department, Addis Ababa University, Addis Ababa, Ethiopia, June 2009.

[10] 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.

[11] C. Dima. Intui2: “A Prototype System for Question Answering over Linked Data,” In Proceedings of the Question Answering over Linked Data lab (QALD-3) at CLEF2013. Lecture Notes in Computer Science (to appear). Springer, 2013.

[12] C. Giannone, V. Bellomaria, and R. Basili. “A Hmm-Based Approach to Question Answering against Linked Data,” In Proceedings of the Question Answering over Linked Data lab (QALD-3) at CLEF2013. Lecture Notes in Computer Science (to appear). Springer, 2013.

[13] S. He, S. Liu, Y. Chen, G. Zhou, K. Liu, and J. Zhao. “Casia@qald-3: A Question Answering System over Linked Data,” In Proceedings of the Question Answering over Linked Data lab (QALD-3) at CLEF2013. Lecture Notes in Computer Science (to appear). Springer, 2013.

[14] C. Pradel, G. Peyet, O. Haemmerle, and N. Hernandez. “Swip at qald-3: Results, Criticisms and Lesson learned,” In Proceedings of the Question Answering over Linked Data lab (QALD-3) at CLEF2013. Lecture Notes in Computer Science (to appear). Springer, 2013.

[15] “ጌታሁን አማረ፣ 1989 የአማርኛ ሰዋስው በቀላል አቀራረብ - Getahun Amare, 1989 Ye-Amarigna Sewasew Beqelal Aqerareb”, Published by 2007 [2014 AD], 'Adis 'Ababa (2007).

[16] ባየ ይማም፣ 1987 የአማርኛ ሰዋስው፣ ት.መ.ማ.ማ.ድ. – Baye Yimam, Ye- AmariGna Sewasew, T.M.M.M.D.

[17] L. Androutsopoulos. “Natural Language Interface to Databases,” an introduction. Journal of Natural Language Engineering, 1(1):2981, 1995.

[18] Xin Li and Dan Roth. “Learning Question Classifiers,” In Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002), pages 556_562, 2002.

[19] Berners-Lee, Tim, James Hendler, and Ora Lassila. "The Semantic Web," Scientific American 284.5 (2001): 28-37.Schilit, Adams, & Want

[20] Christina Unger and André Freitas and Philipp Cimiano, “An Introduction to Question Answering over Linked Data,” M. Koubarakis et al. (Eds.): Reasoning Web 2014, LNCS 8714, pp. 100–140, 2014

[21] Michael L. Brodie and Jason T. Liu. “The power and Limits of Relational Technology in the Age of Information Ecosystems,” Keynote, On the Move Federated Conferences, Heraklion, Greece, October 25-29, 2010, 2011.

[22] Thanh Tran, Tobias Math¨aß, and Peter Haase. “Usability of Keyword-Driven Schema Agnostic Search: A Comparative Study of Keyword Search, Faceted Search, Query Completion and Result Completion,” In Proceedings of the 7th International Conference on The Semantic Web: Research and Applications - Volume Part II, ESWC’10, pages 349–364, Berlin, Heidelberg, 2010. Springer-Verlag.

[23] E. F. Codd. “A Relational Model of Data for Large Shared Data Banks,” Commun. ACM, 13 (6):377–387, June 1970.

[24] Lappoon R. Tang and Raymond J. Mooney. “Using Multiple Clause Constructors in Inductive Logic Programming for Semantic Parsing,” In Proceedings of the 12th European Conference on Machine Learning, pages 466–477, 2001.

[25] Big Data. Wikipedia article, 2014. URL http://en.wikipedia.org/wiki/Big_data, Last accessed on March 05, 2018.

[26] Beat Sprenger. Semantic Crystal. ein end-user-interface zur unterst¨utzung von ontologieabfragen mit sparql, faculty of economics, University of Zurich, 2006.

[27] Semantic Web Stack. Wikipedia article, 2014. URL http://en.wikipedia.org/wiki/Semantic_Web_Stack, Last accessed on April 10, 2018.

[28] Tim Berners-Lee, Linked Data, Web Page, 2006, URL http://www.w3.org/DesignIssues/LinkedData.html, Last accessed on June 10, 2018.

[29] Marcelo Arenas et al. A Direct Mapping of Relational Data to RDF, 2012. URL http://www.w3.org/TR/rdb-direct-mapping/, Last accessed on Sep 21, 2018.

[30] Wikipedia article. Entity Attribute Value Model, 2014. URL http://en.wikipedia.org/wiki/Entity_attribute_value_model, Last accessed on July 11, 2018.

[31] Peshave, Monica, and Kamyar Dezhgosha. "How Search Engines Work: and a Web Crawler Application," Ph.D. diss., University of Illinois Springfield, 2005.

[32] By Yuanlei Zhang, “A Comparison of Search Engines For Finding Resources,” 28, 2004

[33] Valentin Dinu and Prakash M. Nadkarni. “Guidelines for the Effective Use of Entity Attribute Value Modeling for Biomedical Databases,” I. J. Medical Informatics, 76(11 12):769–779, 2007.

[34] RDF Semantic Web Framework, http://www.w3.org/RDF/. Retrieved 9 April 2018.

[35] W3C Consortium Recommendation, http://www.w3.org/TR/. Retrieved 19 October 2018.

[36] Hakia, http://company.hakia.com/whatis.html. Retrieved 23 April 2018.

[37] Vanessa Lopez, Victoria Uren, Marta Sabou, and Enrico Motta. “Is Question Answering for the Semantic Web? A survey,” Semantic Web Journal, 2:125_155, 2011.

[38] Saeedeh Shekarpour, Kemele M. Endris, Ashwini Jaya Kumar, “Question Answering on Linked Data: Challenges and Future Directions,” April 11–15, 2016, Montréal, Québec, Canada. ACM 978-1-4503-4144-8/16/04.

[39] Luke S Zettlemoyer and Michael Collins. “Online Learning of Relaxed CCG Grammars for Parsing to Logical Form,” In EMNLP-CoNLL, pages 678–687, 2007.

[40] Saeedeh Shekarpour, Axel-Cyrille Ngonga Ngomo, and Sören Auer. “Question Answering on Interlinked Data,” In Proceedings of the 22nd International World Wide Web Conference (WWW), pages 11451156, 2013.

[41] Xin Dong and Alon Halevy. “Indexing Dataspaces,” In Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pages 43–54. ACM, 2007.

[42] C. W. Isenberg, “Grammar of The Amharic Language,” London: Richard Watts 1976.

[43] Leslau W (1995),”Reference Grammar of Amharic,” Harrassowitz, Wiesbaden

[44] Imed Zitouni Microsoft Redmond, “WA USA Natural Language Processing of Semitic Languages,” ISBN 978-3-642-45357-1 ISBN 978-3-642-45358-8 (eBook), © Springer-Verlag Berlin Heidelberg 2014

[45] Gasser M (2010), “A Dependency Grammar for Amharic,” In Proceedings of the workshop on language resources and human language technologies for Semitic languages, Valletta

[46] Kramer R (2009), “Definite markers, Phi features, and agreement: a Morphosyntactic Investigation of the Amharic,” DP. PhD thesis, University of California

[47] Kapeliuk O (1988), “Nominalization in Amharic,” Franz Steiner, Stuttgart

[48] H. Keno, "Designing a Named Entity Recognition System for Afaan Oromo Text," MSC, Information Science, Addis Ababa University, Addis Ababa, 2012.

[49] Vanessa Lopez, Andriy Nikolov, Marta Sabou, Victoria S. Uren, Enrico Motta, and Mathieu d’Aquin, “Scaling up Question-Answering to Linked Data,” In Proc. of the 17th International Conference on Knowledge Engineering and Knowledge Management (EKAW), volume 6317 of Lecture Notes in Computer Science, pages 193–210. Springer, 2010.

[50] Vipul Kashyap and Amit Sheth, “Semantic and schematic Similarities between Database Objects: A Context-Based Approach,” The VLDB Journal, 5(4):276–304, December 1996.

[51] David George, “Understanding Structural and Semantic Heterogeneity in the Context of Database Schema Integration,” 2005.

[52] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and S¨oren Auer, “Quality Assessment for Linked Data,” Semantic Web Journal, preprint(preprint), 2015.

[53] Claudio Carpineto and Giovanni Romano, “A Survey of Automatic Query Expansion in Information Retrieval,” ACM Comput. Surv., 44(1):1:1–1:50, January 2012.

[54] C. Unger, L. B□uhmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, P, “Cimiano: SPARQL Template-Based Question Answering,” in: Proc. of the 22nd International World Wide Web Conference (WWW 2012) (2012).

[55] Vanessa Lopez, Miriam Fernández, Enrico Motta, and Nico Stieler, “PowerAqua: Supporting Users in Querying and Exploring the Semantic Web,” Semantic Web, 3(3):249_265, 2012 Web Conference Workshops, volume 7117 of Lecture Notes in Computer Science, pages 125–138. Springer, 2012.

[56] Andr´e Freitas, Dr. Edward Curry, “Treo:Schema-Agnostic Queries for Large-Schema Databases: A Distributional Semantics Approach,” A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy, June 2015

[57] Christina Unger and Philipp Cimiano, “Pythia: Compositional Meaning Construction for Ontology-Based Question Answering on the Semantic Web,” In Natural Language Processing and Information Systems: 16th International Conference on Applications of Natural Language to Information Systems, NLDB 2011, Alicante, Spain, June 28-30, 2011, Proceedings, pages 153_160. Springer, 2011.

[58] A. Kalyanpur et al, “Structured Data and Inference in DeepQA,” IBM Journal of Research & Development, 56(3/4), 2012.

[59] Danica Damljanovic, Milan Agatonovic, and Hamish Cunningham, “Freya: An Interactive Way of Querying Linked Data Using Natural Language,” In Proc. of the European Semantic

[60] Michael Gasser, “HornMorpho: a System for Morphological Processing of Amharic, Oromo, and Tigrinya,” Conference on Human Language Technology for Development, Alexandria, Egypt, 2-5 May 2011.

[61] Zelig Harris. 1954. “Distributional Structure Word,” 10(23):146—162.

[62] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, “Efficient Estimation of Word Representations in Vector Space,” ICLR Workshop, 2013.

[63] Tomas Mikolov, Ilya Sutskever, Kai Chen, “Distributed Representations of Words and Phrases and their Compositionality,” Computation and Language (cs.CL); Machine Learning (cs.LG); Machine Learning (stat.ML), Wed, 16 Oct 2013 23:28:53 UTC

[64] Baroni, M., Dinu, G., & Kruszewski, G. (2014), “Don’t count, Predict! A Systematic Comparison of Context-Counting vs. Context-Predicting Semantic Vectors,” ACL, 238–247. http://doi.org/10.3115/v1/P14-1023

[65] Altszyler, E.; Ribeiro, S.; Sigman, M.; Fernández Slezak, D, "Comparative tudy of LSA vs Word2vec Embeddings in Small Corpora: a Case Study in Dreams Database," arXiv:1610.01520

[66] Fletcher, G.H.L., Van Den Bussche, J., Van Gucht, D., Vansummeren, S, “Towards a Theory of Search Queries,” ACM Trans. Database Syst. 35, 28:1{28:33 (2010)

[67] Karen Spaerck Jones, “A Statistical Interpretation of Term Specificity and its Application in Retrieval,” Journal of Documentation, 28:11–21, 1972.

[68] Kwiatkowski, Tom, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011, “Lexical Generalization in CCG grammar Induction for Semantic Parsing,” In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 1512–1523, Edinburgh, Scotland.

[69] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to Sequence Learning with Neural Networks,” Advances in Neural Information Processing Systems, vol. 4, pp. 3104–3112, 2014.

[70] Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke S. Zettlemoyer, “Scaling Semantic Parsers with On-the-Fly Ontology Matching,” In Empirical Methods in Natural Language Processing, (EMNLP), pages 1545–1556, 2013.

[71] Panupong Pasupat and Percy Liang, “Compositional Semantic Parsing on Semi-Structured Tables,” In ACL(1), 2015.

[72] Jonathan Berant and Percy Liang, “Semantic Parsing via Paraphrasing,” In Annual Meeting for the Association for Computational Linguistics (ACL), 2014.

[73] Siva Reddy, Mirella Lapata, and Mark Steedman, “Large-Scale Semantic Parsing without Question Answer Pairs,” Transactions of the Association for Computational Linguistics, 2:377 392, 2014.

[74] Robin Jia and Percy Liang, “Data Recombination for Neural Semantic Parsing,” In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers, 2016.

[75] Sørensen, T. (1948), "A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species and its Application to Analyses of the Vegetation on Danish Commons," Kongelige Danske Videnskabernes Selskab. 5 (4): 1–34.

[76] Singhal, Amit (2001), "Modern Information Retrieval: A Brief Overview," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering 24 (4): 35–43.

[77] Jianpeng Cheng, Siva Reddy, Vijay Saraswat, Mirella Lapata, “Learning an Executable Neural Semantic Parser,” arXiv:1711.05066v1 [cs.CL] 14 Nov 2017.

[78] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D.Warde-Farley, and Y. Bengio. 2010, “Theano: a CPU and GPU Math Expression Compiler,” In Python for Scientific Computing Conference.

[79] Jason Brownlee, “A Gentle Introduction to Calculating the BLEU Score for Text in Python,” November 20, 2017 in Deep Learning for Natural Language Processing.

[80] Tobias Schnabel, Igor Labutov, David Mimno, “Evaluation Methods for Unsupervised Word Embeddings,” in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 298–307, Lisbon, Portugal, 17-21 September 2015. c 2015 Association for Computational Linguistics.

[81] Daniel, Wayne W. (1990), “Spearman Rank Correlation Coefficient, Applied Nonparametric Statistics (2nd ed.),” Boston: PWS-Kent. pp. 358–365. ISBN 978-0-534-91976-4.

[82] Vanessa Lopeza, Christina Ungerb, Philipp Cimianob, Enrico Mottac, “Evaluating Question Answering over Linked Data,” Preprint submitted to Journal of Web Semantics May 2, 2013.

[83] Evgeniy Gabrilovich and Shaul Markovitch, “Computing Semantic Relatedness using Wikipedia based Explicit Semantic Analysis,” In Proceedings of the 20th International Joint Conference on Artifical Intelligence, IJCAI’07, pages 1606–1611, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[84] Martha Yifiru Tachbelie and Wolfgang Menzel, “Amharic Part-of-Speech Tagger for Factored Language Modeling,” International Conference RANLP 2009 - Borovets, Bulgaria, pages 428–433.

[85] A. A. Argaw and L. Asker, "An Amharic stemmer: Reducing words to their citation forms," in Proceedings of the 2007 workshop on computational approaches to Semitic languages: Common issues and resources, 2007, pp. 104-110.

Annexes

Annex A: Questionnaire form offered to Language Experts to Estimate Semantic Relatedness of Words

We kindly ask you to assist us in a psycholinguistic experiment, aimed at estimating the relatedness of various words in the Amharic language. The purpose of this experiment is to assign relatedness scores to pairs of words, so that the word semantic relatedness evaluator measures how well human perceived relatedness is captured by the word vector representation model; it correlate the distance between word vectors and human perceived semantic relatedness.

Below is a list of pairs of words. For each pair, please assign a numerical relatedness score between 0 and 10 (0 = words are totally unrelated, 10 = words are very closely related). By definition, the relatedness of the word to itself should be 10. You may assign fractional scores (for example, 7.5). The symbols in column 2 (Similarity_type) indicate how pair words are related and give hint to you when you fill their relatedness score; the symbols have the following meaning:

- ✓ i = identical tokens
- ✓ s = synonym (at least in one meaning of each)
- ✓ a = antonyms (at least in one meaning of each)
- ✓ h = first is hyponym of second (at least in one meaning of each)
- ✓ H = first is hyperonym of second (at least in one meaning of each)
- ✓ S = sibling terms (terms with a common hyperonymy)
- ✓ m = first is part of the second one (at least in one meaning of each)
- ✓ M = second is part of the first one (at least in one meaning of each)
- ✓ t = topically related, but none of the above

Specific instructions:

1) The questionnaire starts on the next page.

2) Please fill in your full name at the beginning of the questionnaire. We need the names to ensure individual estimations do not get mixed, and to be able to contact you should any clarifications become necessary.

3) Please fill in the relatedness scores in the appropriate column of the table.

4) If you do not know the meaning of a particular word - please use Amharic dictionary (*መዝገበ ቃላት*), or ask a native Amharic speaker.

5) Please DO NOT consult your friends on assigning the relatedness scores – it is highly important that the scores you assign be independent of someone else's assessment.

6) When estimating relatedness of antonyms, consider them "similar" (i.e., belonging to the same domain or representing features of the same concept), rather than "dissimilar".

Full Name: _____

Table: Questionnaire for Estimation of Words semantic relatedness

No	Similarity_type	Word 1	Word 2	Semantic _relation _score
1	H	ተጨዋች	አጥቂ	
2	t	ቡድን	ተጨዋች	
3	t	ቀን	ክረምት	
4	t	ተጨዋች	አሰልጣኝ	
5	i	ቡድን	ቡድን	
6	t	ሚስት	ባል	
7	t	ወንድ	ሴት	
8	h	ሶማሌያ	ክልል	
9	M	አገር	ክልል	
10	t	ትምህርት	ተማሪ	
11	t	ዘመን	አመት	
12	s	ገንዘብ	ብር	
13	t	ገንዘብ	ባንክ	
14	h	ኢትዮጵያ	አገር	
15	H	ቢዝነስ	ፓለቲካ	
16	h	ሙዚቃ	መዝናኛ	
17	s	ሙዚቀኛ	ዘፋኝ	
18	S	አርጋን	ቫዮሊን	
19	S	ዳንሰል	ሂፖፕ	
20	h	ነሀስ	ሜዳሊያ	
21	S	ነሀስ	ወርቅ	
22	t	አክስት	አጐት	
23	s	ሰንደቅ	ባንዲራ	
24	m	ቀበሌ	ወረዳ	
25	H	አትሌት	ቀነኔሳ	
26	H	ቋንቋ	ስልጢኛ	
27	t	ፍርድ	ችሎት	
28	t	ይግባኝ	ችሎት	
29	t	ደስታ	ሀዘን	
30	M	ኢትዮጵያ	አፍሪካ	
31	S	ሀምሌ	ነሀሴ	
32	h	ማንጐ	ጭማቂ	
33	t	ሳይንቲስት	ምጥቅ	
34	t	አርሰናል	አንፊልድ	
35	t	ሽበት	ሂና	
36	t	ሰማያዊ	ፓርቲ	
37	t	ምስክር	ወረቀት	
38	h	አርቶዶክስ	ክርስቲያን	
39	t	ሜትር	ርዝመት	
40	S	ምስራቅ	ምእራብ	
41	t	ልጅ	አባት	

42	t	እናት	አባት	
43	t	ቧንቧ	ውሀ	
44	H	ትምባሆ	ሮዝማን	
45	t	ሀውልት	መታሰቢያ	
46	H	ቢራ	ዳሸን	
47	h	ኤፍኤም	ሬዲዮ	
48	t	ዩኒቨርስቲ	ቅድመምረቃ	
49	t	ኢንተርን	ሀኪም	
50	S	ጫት	ሺሻ	
51	t	አበምኔት	ገዳም	
52	t	ድርቅ	ረሀብ	
53	t	ጽድቅ	ኩነኔ	
54	t	አናሎግ	ዲጂታል	
55	t	ዲያቆን	ቤተክርቲያን	
56	t	ዘረኝነት	እብሪተኛነት	
57	t	ሙሰኛ	ትምክህትኛ	
58	t	ጁመአ	መድረሳ	
59	H	ውቅያኖስ	ሜድትራኒያ	
60	t	ሊቃውንት	አመስጥረ	
61	t	ዋተተ	አመነጨ	
62	S	ቅንነት	ታማኝነት	
63	t	ትእይንት	ሚራክል	
64	t	ነጐድግዋድ	ጨለማ	
65	t	ቅርስ	ዩኔስኮ	
66	t	ፊርማ	ማህተም	
67	h	ቶምቦላ	ሎተሪ	
68	t	ብጹአን	ሲኖዶስ	
69	t	ወይን	ጠጅ	
70	H	ሹም	ኤታማጆር	
71	t	አክራሪ	ጽንፈኛ	
72	M	ሬስቶራንት	ሜኑ	
73	S	ድንች	ጐመን	
74	m	ሙሽራ	ሰርግ	
75	t	ስነምግባር	ጸረሙስና	
76	S	ትዝታ	አንቺሆይ	
77	t	ቪዛ	ፓስፖርት	
78	t	ቅቤ	አይብ	
79	h	ግእዝ	ፊደላት	
80	s	ሞባይል	ስልክ	
81	S	አማርኛ	ኦሮሚፋ	
82	t	ነዳጅ	ቤንዚን	
83	H	ወንዝ	ጤግሮስ	
84	h	ሺናሻ	ብሄረሰብ	
85	t	ኢንጅነር	አማካሪ	
86	S	ቴሌቪዥን	ሬድዮን	
87	t	ቴአትር	ተውኔት	
88	t	ስትራቴጂ	ፖሊሲ	
89	t	ዘይት	አቸቶ	
90	h	ፖታሽ	ማእድን	
91	t	አባይ	ጣና	
92	t	ማኔጅመንት	ፐርቼዚንግ	
93	t	አርሻ	ፖኬጅ	
94	S	ሆሊውድ	ቦሊውድ	
95	t	ታይፎይድ	ታይፈስ	
96	t	ጅምናስቲክ	አክሮባት	
97	t	ወረኛ	አድመኛ	
98	s	ቂመኛ	በቀለኛ	
99	s	አጨብጫቢ	አጫፋሪ	
100	m	ህዝብ	አገር	

Annex B: Human's mean semantic relatedness score and scores obtained from different semantic Embedding models.

No	Similarity_type	Word 1	Word 2	WVM with Stemmed	WVM not Stemmed	WVM not Sampled	Human mean Semantic_relation_score	Ur_1	Ur_2	Ur_3	Ur_4	Ur_5	Ur_6	Ur_7	Ur_8	Ur_9	Ur_10
1	H	ተጨዋች	አጥቂ	0.65	0.6	0.69	0.63	7	5	6	7	2	5	8	8	8	7
2	t	ቡድን	ተጨዋች	0.65	0.61	0.69	0.64	6	7	5	7	8	8	6	6	5	6
3	t	ቀን	ክረምት	0.37	0.29	0.31	0.38	4	5	0	5	4	6	5	4	0	5
4	t	ተጨዋች	አሰልጣኝ	0.65	0.63	0.59	0.56	6	9	5	7	8	6	6	2	0	7
5	i	ቡድን	ቡድን	1.0	1	1.0	1.0	10	10	10	10	10	10	10	10	10	10
6	t	ሚስት	ባል	0.61	0.57	0.59	0.65	7	6	4	8	9	5	9	5	7	5
7	t	ወንድ	ሴት	0.78	0.7	0.77	0.72	7	6	8	8.5	9	5	8	7	7	6
8	h	ሶማሌያ	ክልል	0.66	0.58	0.66	0.61	8	7	6.5	7.5	1	7	6	5	4	9
9	M	አገር	ክልል	0.58	0.57	0.52	0.65	6.5	5	5.5	7	3	7	6	7	9	9
10	t	ትምህርት	ተማሪ	0.67	0.59	0.69	0.66	8	8	3	6	8	7	6	5	7	8
11	t	ዘመን	አመት	0.61	0.56	0.64	0.63	7.5	7	4.5	6.5	8	6	7	7	5	4
12	s	ገንዘብ	ብር	0.71	0.68	0.69	0.75	8	7	7	9	9	7	8	9	7	4
13	t	ገንዘብ	ባንክ	0.63	0.64	0.67	0.70	8	8	4	7	7	7	8	8	5	8
14	h	ኢትዮጵያ	አገር	0.74	0.72	0.78	0.80	9	7	7	8	9	6	8	8	9	9
15	H	ቢዝነስ	ፓለቲካ	0.47	0.56	0.64	0.50	5	2	8	6	5	5	7	4	5	3
16	h	ሙዚቃ	መዝናኛ	0.74	0.64	0.72	0.68	7	8	1.5	7	8	8	7	7	5	9
17	s	ሙዚቀኛ	ዘፋኝ	0.76	0.69	0.75	0.80	9	8	7	9	9	6	7	9	8	8

18	S	አርጋን	ቫዮሊን	0.65	0.61	0.68	0.63	8	7	6	8	7	8	7	4	0	8
19	S	ዳንሶል	ሂፖፕ	0.76	0.67	0.76	0.71	5	7	9	8	7	8	7	5	7	8
20	h	ነሀስ	ሜዳሊያ	0.77	0.68	0.73	0.71	6	8	8	7.5	7	8	6	6	5	9
21	S	ነሀስ	ወርቅ	0.57	0.47	0.58	0.65	3	8	5	8	6	8	6	7	5	9
22	t	አክስት	አጐት	0.59	0.45	0.53	0.62	7	7	2	8.5	8	7	6	4	6	6
23	s	ሰንደቅ	ባንዲራ	0.68	0.73	0.56	0.79	8	7	8	9	9	7	8	8	7	8
24	m	ቀበሌ	ወረዳ	0.71	0.68	0.75	0.73	7.5	7	9	8	8	7	6	6	6	8
25	H	አትሌት	ቀነኒሳ	0.63	0.59	0.67	0.67	7	8	7	8	8	6	5	7	5	6
26	H	ቋንቋ	ስልጢኛ	0.69	0.6	0.69	0.65	7	7	6	7	9	7	5	6	7	4
27	t	ፍርድ	ችሎት	0.61	0.59	0.74	0.79	9	8	3.5	9	8	9	7	9	9	7
28	t	ይግባኝ	ችሎት	0.64	0.64	0.56	0.69	8	9	3	7.5	8	6	7	5	7	8
29	t	ደስታ	ሀዘን	0.4	0.35	0.49	0.42	4	3	2	8	9	4	3	2	3	4
30	M	ኢትዮጵያ	አፍሪካ	0.54	0.54	0.62	0.47	4	2	4	7.5	5	4	5	4	2	9
31	S	ሀምሌ	ነሀሴ	0.59	0.57	0.57	0.64	6	9	6	8	8	5	6	6	4	6
32	h	ማንጐ	ጭማቂ	0.6	0.55	0.63	0.63	5	7	6	7	5	4	6	7	7	9
33	t	ሳይንቲስት	ምጥቅ	0.67	0.66	0.6	0.75	7	8	7.5	6	9	8	7	5	9	8
34	t	አርሰናል	አንፊልድ	0.48	0.53	0.61	0.47	4	3	2	7	8	8	3	2	3	7
35	t	ሽበት	ሂና	0.54	0.5	0.53	0.48	3	6	4	7	4	5	2	6	5	6
36	t	ሰማያዊ	ፓርቲ	0.52	0.6	0.69	0.46	2	4	5	6	2	7	4	5	4	7
37	t	ምስክር	ወረቀት	0.59	0.55	0.56	0.53	7	1	4	6	4	7	7	5	5	7
38	h	አርቶዶክስ	ክርስቲያን	0.65	0.51	0.67	0.71	8	5	7	7	9	5	7	9	6	8
39	t	ሜትር	ርዝመት	0.64	0.65	0.67	0.60	8	7	5	7	9	8	5	2	3	6
40	S	ምስራቅ	ምእራብ	0.59	0.62	0.63	0.56	8	3	7	8	9	9	5	1	0	6
41	t	ልጅ	አባት	0.65	0.57	0.69	0.64	7	8	5	8	9	7	5	1	8	6
42	t	እናት	አባት	0.58	0.59	0.61	0.67	8	6	5	8	9	6	5	6	8	6
43	t	ቧንቧ	ውሀ	0.6	0.53	0.63	0.58	8	9	1	7.5	6	5	8	2	5	6
44	H	ትምባሆ	ሮዝማን	0.67	0.45	0.65	0.71	8	7	7	7.	7	6	8	7	6	7

45	t	ሀውልት	መታሰቢያ	0.65	0.51	0.65	0.69	7	8	4	5 8	6	7	6	6	9	8
46	H	ቢራ	ዳሸን	0.66	0.49	0.65	0.62	6	7	8	7.5	8	5	5	5	4	6
47	h	ኤፍኤም	ሬዲዮ	0.63	0.54	0.63	0.71	8.5	6	9.5	7.5	9	7	5	5	7	6
48	t	ዩኒቨርስቲ	ቅድመምረቃ	0.66	0.56	0.68	0.63	8	9	5	7	8	5	4	6	5	6
49	t	ኢንተርን	ሀኪም	0.63	0.48	0.65	0.63	8	7	5	7.5	8	9	5	4	4	5
50	S	ጫት	ሺሻ	0.74	0.57	0.64	0.69	7	2	7	7	8	9	7	5	9	8
51	t	አበምኔት	ገዳም	0.67	0.68	0.67	0.61	6	7	6	7.5	8	6	9	2	5	4
52	t	ድርቅ	ረሀብ	0.65	0.64	0.6	0.72	8	8	4	8.5	9	9	9	7	4	5
53	t	ጽድቅ	ኩነኔ	0.51	0.55	0.62	0.47	8	3	5	6	7	5	3	1	5	4
54	t	አናሎግ	ዲጂታል	0.61	0.52	0.64	0.53	8	3	4	8.5	9	4	6	1	4	5
55	t	ዲያቆን	ቤተክርቲያን	0.66	0.58	0.64	0.67	8	9	7	7.5	7	5	8	7	3	5
56	t	ዘረኝነት	እብሪተኛነት	0.64	0.46	0.59	0.52	5	7	4.5	4	1	8	7	5	5	5
57	t	ሙሰኛ	ትምክህትኛ	0.46	0.38	0.59	0.40	4	5	2	5	3	8	4	2	1	6
58	t	ጁመአ	መድረሳ	0.57	0.56	0.55	0.51	7	2	4	8	6	7	4	3	3	7
59	H	ውቅያኖስ	ሜድትራኒያ	0.62	0.54	0.58	0.66	6.5	9	5	7	8	7	7	3	7	6
60	t	ሊቃውንት	አመስጥረ	0.65	0.54	0.61	0.62	7	8	3.5	7	7	6	7	4	5	7
61	t	ዋተተ	አመነጨ	0.49	0.59	0.47	0.45	6	7	5	4	3	6	2	1	4	7
62	S	ቅንነት	ታማኝነት	0.62	0.58	0.67	0.69	8.5	6	6	7	8	7	7	7	6	6
63	t	ትእይንት	ሚራክል	0.55	0.47	0.6	0.49	7	4	4	5	1	6	4	5	5	8
64	t	ነጎድጓዋድ	ጨለማ	0.52	0.43	0.53	0.48	6	8	4	3	2	8	4	2	4	7
65	t	ቅርስ	የኔስኮ	0.66	0.65	0.65	0.60	8	8	4	7.	8	7	5	1	5	6

66	t	ፊርማ	ማህተም	0.58	0.34	0.52	0.66	7.5	7	8	5.7	8	8	6	5	5	4
67	h	ቶምቦላ	ሎተሪ	0.63	0.63	0.67	0.70	8	5	7.5	8	8	7	6	6	9	5
68	t	ብጹአን	ሲኖዶስ	0.59	0.55	0.64	0.61	9	8	7	6.5	7	6	6	4	3	4
69	t	ወይን	ጠጅ	0.67	0.44	0.69	0.66	7	9	6	8	8	8	6	5	3	6
70	H	ሹም	ኤታማጆር	0.55	0.59	0.57	0.63	6	7	5	7	7	8	7	4	5	7
71	t	አከራሪ	ጽንፈኛ	0.74	0.59	0.71	0.76	8	9	10	7	9	8	7	6	5	7
72	M	ሬስቶራንት	ሜኑ	0.61	0.54	0.64	0.59	5	8	5	6.5	8	5	7	2	6	6
73	S	ድንች	ጐመን	0.62	0.57	0.64	0.61	8.5	5	5	7	5	8	5	4	5	8
74	m	ሙሽራ	ሰርግ	0.72	0.6	0.74	0.68	9	6	7	8	9	6	8	2	6	7
75	t	ስነምግባር	ጸረሙስና	0.56	0.45	0.55	0.61	6	8	5	6.5	5	7	8	4	6	5
76	S	ትዝታ	አንቺሆይ	0.55	0.54	0.6	0.62	6	7	6	7	7	6	7	5	5	6
77	t	ቪዛ	ፓስፖርት	0.63	0.56	0.69	0.70	7	8	7.5	8	9	5	7	9	3	6
78	t	ቅቤ	አይብ	0.64	0.62	0.68	0.73	8	9	9	7.5	8	6	7	8	3	7
79	h	ግእዝ	ፊደላት	0.73	0.58	0.65	0.68	8	8	5.5	6	7	8	7	8	5	5
80	s	ሞባይል	ስልክ	0.68	0.55	0.64	0.74	7	8	7	8	8	8	5	9	5	9
81	S	አማርኛ	ኦሮሚፋ	0.67	0.44	0.65	0.68	7.5	9	6	8	7	7	5	4	6	8
82	t	ነዳጅ	ቤንዚን	0.69	0.53	0.65	0.66	7	8	0.5	9	8	6	6	9	5	7
83	H	ወንዝ	ጤግሮስ	0.67	0.56	0.69	0.74	6	7	9	6	8	8	7	9	6	8
84	h	ሺናሻ	ብሄረሰብ	0.73	0.57	0.7	0.65	4	6.5	9.5	7	8	7	6	7	3	7
85	t	ኢንጅነር	አማካሪ	0.65	0.42	0.6	0.55	0	6	8	6	3	8	5	4	8	7
86	S	ቴሌቪዥን	ሬድዮን	0.67	0.44	0.65	0.61	3	6	7	8	8	6	7	5	5	6
87	t	ቴአትር	ተውኔት	0.64	0.4	0.69	0.73	6	9	4.	9	8	6	7	9	7	7

88	t	ስትራቴጂ	ፖሊሲ	0.48	0.25	0.72	0.62	7	8	5 4	3	8	5	7	8	5	7
89	t	ዘይት	አቸቶ	0.55	0.45	0.61	0.47	6	7	4	7	2	4	5	2	3	7
90	h	ፖታሽ	ማእድን	0.71	0.56	0.76	0.69	5	8	7	7	8	7	7	6	8	6
91	t	አባይ	ጣና	0.57	0.54	0.51	0.61	8	5	4. 5	8	8	4	6	7	5	5
92	t	ማኔጅመንት	ፐርቼዚንግ	0.66	0.5	0.57	0.53	6	8	4	7	3	7	6	2	4	6
93	t	አርሻ	ፖኬጅ	0.63	0.69	0.63	0.52	6	6	3	5	2	7	5	4	7	7
94	S	ሆሊውድ	ቦሊውድ	0.66	0.48	0.67	0.70	6. 5	9	7	7. 5	7	6	7	6	6	8
95	t	ታይፎይድ	ታይፈስ	0.8	0.65	0.83	0.72	7	8	7	8	6	8	6	5	9	8
96	t	ጅምናስቲክ	አክሮባት	0.54	0.33	0.65	0.71	8	9	6	9	6	4	6	8	9	6
97	t	ወረኛ	አድመኛ	0.66	0.52	0.62	0.52	4	6	6	5	2	6	6	2	8	7
98	s	ቂመኛ	በቀለኛ	0.61	0.67	0.73	0.78	8	9	7	7	8	7	8	8	7	9
99	s	አጨብጫቢ	አጫፋሪ	0.65	0.52	0.65	0.73	6	9	5	7	8	7	9	8	5	9
100	m	ህዝብ	አገር	0.67	0.6	0.61	0.71	6	5	7	7	7	8	7	7	8	9

Annex C: All indivisual BLEU scores for each test question of NSPM with Stemmed model

Que ID	Question	Target	Predicted	BLEU-1	BLEU-2	BLEU-3	BLEU-4
1	በሀረሪ ክልል ውስጥ የሚገኙትን ሀይቆች ስንት ናቸው	answer (count (A) , ሀይቅ (A), መገኛ_ክልል (A , ሀረሪ_ክልል))	answer (count (A) , ሀይቅ (A), መገኛ_ክልል (A , ሀረሪ_ክልል))	1.0	1.0	1.0	1.0
2	ኦሮሚያ ክልል ውስጥ ሸናሻ ይነገራል	answer (? , የሚነገር_ቋንቋ (ኦሮሚያ_ክልል , ሸናሻ))	answer (? , የሚነገር_ቋንቋ (ኦሮሚያ_ክልል , ሸናሻ))	1.0	1.0	1.0	1.0
3	አዳማ የሚገኘው ስታድየም የተገነባው መቸ ነው	answer (A , ስታድየም (B) , መገኛ_ቦታ (B , አዳማ) , ግንባታ_ቀን (B , A))	answer (A , መገኛ_ቦታ (A , አዳማ))	0.34	0.30	0.25	0.15
4	አዲስ አበባ የሚገኙት ስታድየም ስንት ናቸው	answer (count (A) , ስታድየም (B) , መገኛ_ቦታ (A , አዲስ_አበባ))	answer (count (A) , መገኛ_ቦታ (A , አዲስ_አበባ))	0.70	0.70	0.70	0.70
5	ማሞ ወልዴ ባጠቃላይ ስንት ብር አስገኝቷል	answer (A , የተመዘገበ_ብር (ማሞ_ወልዴ , A))	answer (A , የተመዘገበ_ብር (ማሞ_ወልዴ , A))	1.0	1.0	1.0	1.0
6	የኢትዮጵያ የሴቶች ብሄራዊ ቡድን መሀል አጥቂ ስንት ናቸው	answer (count (A) , ተጨዋች (ኢትዮጵያ_ሴቶች_ብሄራዊ_ቡድን , A) , ጨዋታ_ቦታ (A , መሀል_አጥቂ))	answer (count (A) , ተጨዋች (ኢትዮጵያ_ሴቶች_ብሄራዊ_ቡድን , A))	0.61	0.61	0.59	0.58
7	በቤንሻንጉል ጉሙዝ ክልል የሚገኙትን ሀይቆች ጥቀስ	answer (A , ሀይቅ (A), መገኛ_ክልል (A , ቤንሻንጉል_ጉሙዝ_ክልል))	answer (A , ሀይቅ (A), መገኛ_ክልል (A , ቤንሻንጉል_ጉሙዝ_ክልል))	1.0	1.0	1.0	1.0
8	ጉና ተራራ የሚገኘው ሀረሪ	answer (? , መገኛ_ክልል	answer (? , መገኛ_ክልል (	1.0	1.0	1.0	1.0

	ክልል ነው	(ጉና_ተራራ , ሀረሪ_ክልል))	ጉና_ተራራ , ሀረሪ_ክልል))				
9	ቢንያም ደምጤ የሚጫወተው መሀል አጥቂ ቦታ ላይ ነው	answer (? , እግርኳስ_ተጨዋች (ቢንያም_ደምጤ) , ጨዋታ_ቦታ (ቢንያም_ደምጤ , መሀል_አጥቂ))	answer (? , እግርኳስ_ተጨዋች (ቢንያም_ደምጤ) , ጨዋታ_ቦታ (ቢንያም_ደምጤ , መሀል_አጥቂ))	1.0	1.0	1.0	1.0
10	በኢትዮጵያ የሚገኘው ረዝሙ ተራራ ማነው	answer (A , argmax ((ተራራ (A), መገኛ_ሀገር (A , ኢትዮጵያ) , ከፍታ (A , B)) , B))	answer (A , argmax ((ተራራ (A), መገኛ_ሀገር (A , ኢትዮጵያ) , ከፍታ (A , B))	0.86	0.86	0.86	0.86
11	የእጅጋዬሁ ዲባባ እህቶች ባጠቃላይ ስንት ብር አስመዝገቡ	answer (count (A) , የተመዘገበ_ብር (B , A) , እህት (B , እጅጋዬሁ_ዲባባ))	answer (count (A) , የተመዘገበ_ነህስ (B , እህት (B , እህት (B , እህት (B , እህት (B , እህት (B , እህት (B , እህት (B , እህት	0.15	0.13	0.15	0.10
12	ጥሩነሽ ዲባባ ከሴት እሯጮች በወርቅ ብዛት ትበልጣለች	answer (? , የተመዘገበ_ወርቅ (ጥሩነሽ_ዲባባ , B) , የተመዘገበ_ወርቅ (A , C) , እሯጭ (A) , ይበልጣል (B , C))	answer (? , የተመዘገበ_ወርቅ (ጥሩነሽ_ዲባባ , የተመዘገበ_ወርቅ (ጥሩነሽ_ዲባባ , B))	0.34	0.33	0.31	0.28
13	አበበ ቢቂላ የ5000 ሜትር እሯጭ ነው	answer (? , የተሳተፈበት_ሩጫ_አይነት (አበበ_ቢቂላ , 5000_ሜትር) , እሯጭ (አበበ_ቢቂላ))	answer (? , የተሳተፈበት_ሩጫ_አይነት (አበበ_ቢቂላ , 5000_ሜትር) , እሯጭ (አበበ_ቢቂላ))	1.0	1.0	1.0	1.0
14	እጅጋዬሁ ዲባባ ስንት እሯጭ እህቶች አሏት	answer (count (A) , እህት (A , እጅጋዬሁ_ዲባባ) , እሯጭ (A))	answer (count (A) , እህት (A , እህት (A , እህት (A , እህት (A , እህት (A , እህት	0.14	0.12	0.14	0.10
15	መሰረት ደፋር ምን ምን እሩጫ ላይ ተሳተፉለች	answer (A , የተሳተፈበት_ሩጫ_አይነት (መሰረት_ደፋር , A))	answer (A , የተሳተፈበት_ሩጫ_አይነት (መሰረት_ደፋር , A))	1.0	1.0	1.0	1.0

16	የፊት መስመር እግርኳስ ተጨዋቾችን ጥቀስልኝ	answer (A , እግርኳስ_ተጨዋች (A) , ጨዋታ_ቦታ (A , አጥቂ))	answer (A , እግርኳስ_ተጨዋች (A , አጥቂ))	0.63	0.63	0.63	0.61
17	ከደራርቱ ቱሉ እና ከገንዘቤ ዲባባ በሜዳሊያ ብዛት ማን ያንሳል	answer (who , ሜዳሊያ (ደራርቱ_ቱሉ , A) , ሜዳሊያ (ገንዘቤ_ዲባባ , B) , ያንሳል (count (A) , count (B)))	answer (who , ሜዳሊያ (ገንዘቤ_ዲባባ , A) , count (B)) , ያንሳል (count (A) , count (B)) , ያንሳል (count (A) , count (B)) , ያንሳል (count (A	0.29	0.27	0.28	0.23
18	ደደቢት እግርኳስ ክለብ ዋንጫ በልቶ ያውቃል	answer (? , ዋንጫ_ብዛት (ደደቢት_እግርኳስ_ክለብ , A))	answer (? , ዋንጫ_ብዛት (ደደቢት_እግርኳስ_ክለብ_ፉትቦል_ክለብ , ዋንጫ_ብዛት (ደደቢት_እግርኳስ_ክለብ_ክለብ , A))	0.67	0.62	0.59	0.49
19	ቀነኒሳ በቀለ ስንት የሩጫ አይቶች ላይ ተሳተፈ	answer (count (A) , የተሳተፈበት_ሩጫ_አይነት (ቀነኒሳ_በቀለ , A))	answer (count (A) , የተሳተፈበት_ሩጫ_አይነት (ቀነኒሳ_በቀለ , A))	1.0	1.0	1.0	1.0
20	አዳማ የሀረሪ ክልል ዋና ከተማ ነው	answer (? , ዋና_ከተማ (ሀረሪ_ክልል , አዳማ))	answer (? , ዋና_ከተማ (ሀረሪ_ክልል , አዳማ))	1.0	1.0	1.0	1.0
21	የሲዳማ ቡና ቅጽል ስም ማን ይባላል	answer (A , ቅጽል_ስም (ሲዳማ_ቡና , A))	answer (A , ቅጽል_ስም (ሲዳማ_ቡና , A))	1.0	1.0	1.0	1.0
22	አባይ ሀይቅ በስፋት ከጣና ሀይቅ ያንሳል	answer (? , ስፋት(አባይ_ሀይቅ , A) , ስፋት(ጣና_ሀይቅ, B) , ያንሳል (A , B))	answer (? , ስፋት(ጣና_ሀይቅ, B) , ያንሳል (A , B))	0.69	0.69	0.69	0.67
23	በቤንሻንጉል ጉሙዝ ክልል የሚገኙትን ታሪካዊ ቦታውችን ዘርዝር	answer (A , ታሪካዊ_ቦታ (A) , መገኛ_ክልል (A , ደቡብ_ብሄር_ብሄረሰቦች_ክልል))	answer (A , ታሪካዊ_ቦታ (A) , ደቡብ_ብሄር_ብሄረሰቦች_ክልል))	0.72	0.72	0.69	0.66
24	ሶማሌ ክልል ውስጥ የሚነገሩ ቋንቋዎችን ስንት ናቸው	answer (count (A) , የሚነገር_ቋንቋ (ሶማሌ_ክልል , A))	answer (count (A) , የሚነገር_ቋንቋ (ሶማሌ_ክልል , A))	1.0	1.0	1.0	1.0

25	ኡመድ ኡክሪ የተጫወተባቸው ክለቦችን ጥቀስ	answer (A , የሚጫወትበት_ቡድን (ኡመድ_ኡክሪ , A))	answer (A , የሚጫወትበት_ቡድን (ኡመድ_ኡክሪ , A))	1.0	1.0	1.0	1.0
26	ዘንገና ሀይቅ ንጹህ ውሀ ሀይቅ ነው	answer (? , ሀይቅ_አይነት (ዘንገና_ሀይቅ , ንጹህ_ውሀ_ሀይቅ))	answer (? , ሀይቅ_አይነት (ዘንገና_ሀይቅ , ንጹህ_ውሀ_ሀይቅ))	1.0	1.0	1.0	1.0
27	በርታ የት ክልል ነው የሚነገረው	answer (A , ክልል (A) , የሚነገር_ቋንቋ (A , በርታ))	answer (A , ክልል (A) , የሚነገር_ቋንቋ (A , በርታ))	1.0	1.0	1.0	1.0
28	ጭላሎ ተራራ የሚገኘው ቤንሻንጉል ጉሙዝ ክልል ነው	answer (? , መገኛ_ክልል (ጭላሎ_ተራራ , ቤንሻንጉል_ጉሙዝ_ክልል))	answer (? , መገኛ_ክልል (ጉና_ተራራ , ቤንሻንጉል_ጉሙዝ_ክልል))	0.91	0.85	0.80	0.70
29	አሮሚያ ክልል የሚገኙትን ሀይቅ ጥቀስ	answer (A , ሀይቅ (A), መገኛ_ክልል (A , አሮሚያ_ክልል))	answer (A , ሀይቅ (A), መገኛ_ክልል (A , አሮሚያ_ክልል))	1.0	1.0	1.0	1.0
30	በሀረሪ ክልል ከሚገኙት ተራሮች መካከል በርዝመት ትንሹ ተራራ ከፍታው ስንት ነው	answer (A , argmin ((ተራራ (B), መገኛ_ክልል (B , ሀረሪ_ክልል) , ከፍታ (B , A)) , A))	answer (A , argmin ((ተራራ (B), መገኛ_ክልል (B),) , ከፍታ (B), መገኛ_ክልል (B),) , ከፍታ (B), መገኛ_ክልል (B),) , ከፍታ (B	0.26	0.22	0.23	0.18
31	ከአፋር ክልል እና ከትግራይ ክልል በተራራ ብዛት ማን ይበልጣል	answer (who , መገኛ_ክልል (A , አፋር_ክልል) , መገኛ_ክልል (B , ትግራይ_ክልል) , ይበልጣል (count (A) , count (B)))	answer (who , መገኛ_ክልል (A , ሶማሌ_ክልል) , መገኛ_ክልል (B , ትግራይ_ክልል) , ይበልጣል (count (A) , count (B)))	0.97	0.95	0.94	0.91
32	የጋምቤላ ክልል ተወላጅ የሆኑትን እሯጮችን ዘርዝር	answer (A , ክልል (A , አሮሚያ) , እሯጭ (A))	answer (A , ክልል (A , አሮሚያ) , እሯጭ (A))	1.0	1.0	1.0	1.0
33	የአፋር ክልል በህዝብ ብዛት ከትግራይ ክልል ይበልጣል	answer (? , ህዝብ_ብዛት(አፋር_ክልል , A) , ህዝብ_ብዛት(ትግራይ_ክልል , B) ,	answer (? , ህዝብ_ብዛት(አሮሚያ_ክልል , A))	0.25	0.23	0.20	0.15

		ይበልጣል (A , B))					
34	የትኛው የክልል ዋና ከተማ ነው ትንሽ የህዝብ ቁጥር ያለው	answer (A , argmin ((ዋና_ከተማ (B , A) , ህዝብ_ብዛት (A , C)) , C))	answer (A , argmin ((ዋና_ከተማ (A) , ህዝብ_ብዛት (A , C))	0.73	0.71	0.69	0.66
35	ትግራይ ስታድየም ተመልካች በመያዝ አቅም ከአድስ አበባ ስታድየም ይበልጣል	answer (? , ተመልካች_አቅም (ትግራይ_ስታድየም , A) , ተመልካች_አቅም (አድስ_አበባ_ስታድየም , B) , ይበልጣል (A , B))	answer (? , ተመልካች_አቅም (ትግራይ_ስታድየም , B))	0.28	0.28	0.27	0.25
36	የፋሲል ከነማ ስፖርት ክለብ መቸ ተመሰረተ	answer (A , ምስረታ_ቀን (ፋሲል_ከነማ_ክለብ , A))	answer (A , ምስረታ_ቀን (ፋሲል_ከነማ_ክለብ , A))	1.0	1.0	1.0	1.0
37	በርዝመት ትልቁ ተራራ የት ክልል ውስጥ ይገኛል	answer (A , argmax ((ተራራ (B) , መገኛ_ክልል (B , A) , ከፍታ (B , C)) , C))	answer (A , argmax ((ተራራ (B , A))	0.32	0.32	0.31	0.30
38	ሀይሌ ገብረስላሴ የት ነው የተወለደው	answer (A , ትውልድ_ቦታ (ሀይሌ_ገብረስላሴ , A))	answer (A , ትውልድ_ቦታ (ሀይሌ_ገብረስላሴ , A))	1.0	1.0	1.0	1.0
39	ብዛት ዋንጫ የበላው ፉትቦል ቡድን ማነው	answer (A , argmax ((ፉትቦል_ክለብ (A) , ዋንጫ_ብዛት (A , B)) , B))	answer (A , argmax (count(, A))	0.30	0.27	0.23	0.19
40	ደራርቱ ቱሉ ያገኘችውን ሽልማቶች ተናገር	answer (A , ሽልማት (ደራርቱ_ቱሉ , A))	answer (A , ሽልማት (ደራርቱ_ቱሉ , A))	1.0	1.0	1.0	1.0
41	ለኢትዮጵያ ብሄራዊ ቡድን ብዙ ጎል ያስገባው ተጨዋች ማነው	answer (A , argmax ((ተጨዋች (ኢትዮጵያ_ብሄራዊ_ቡድን , A) , ጎል_ብዛት (A , B)) , B))	answer (A , argmin (ኢትዮጵያ_ብሄራዊ_ቡድን , A))]	0.25	0.24	0.21	0.17
42	ሲሳይ አብርሀም የማን እግርኳስ ክለብ ሊቀመንበር ነው	answer (A , ሊቀመንበር (A , ሲሳይ_አብርሀም) , ፉትቦል_ክለብ (A))	answer (A , ሊቀመንበር (A , ሲሳይ_አብርሀም) , ፉትቦል_ክለብ (A))	1.0	1.0	1.0	1.0

43	ከወንድ እሯጮች መካከል ብዙ ሜዳልያ ያስመዘገበው ማነው	answer (A , argmax ((ጾታ (A , ወንድ) , እሯጭ (A) , ሜዳሊያ (A , B)) , count (B)))	answer (A , argmin ((ጾታ (A) , ሜዳሊያ (A , ኢትዮጵያ) , እሯጭ (A))	0.63	0.58	0.54	0.48
44	1500 ሜትር እሩጫ ላይ የተሳተፉ የኢትዮጵያ እሯጮችን ዘርዝር	answer (A , የተሳተፈበት_ሩጫ_አይነት (A , 1500_ሜትር) , እሯጭ (A))	answer (A , የተሳተፈበት_ሩጫ_አይነት (A , 1500_ሜትር) , እሯጭ (A))	1.0	1.0	1.0	1.0
45	በኦሮሚያ ክልል የሚገኘው በረዝመት ትንሹ ተራራ ማነው	answer (A , argmin ((ተራራ (A), መገኛ_ክልል (A , ኦሮሚያ_ክልል) , ከፍታ (A , B)) , B))	answer (A , argmin ((ተራራ (A), መገኛ_ክልል (A , ኦሮሚያ_ክልል) , ከፍታ (A , ኦሮሚያ_ክልል) , ከፍታ (A , ኦሮሚያ_ክልል	0.27	0.24	0.27	0.22
46	ታሪኩ በቀለ የት ነው የተወለደው	answer (A , ትውልድ_ቦታ (ታሪኩ_በቀለ , A))	answer (A , ትውልድ_ቦታ (ታሪኩ_በቀለ , A))	1.0	1.0	1.0	1.0
47	የቴዲ አፍሮ ያስተሰርያል የሚለው የአልበም ስም በውስጡ ስንት ዘፈኖች አሉት	answer (count (A) , ዘፋኝ (ቴዲ_አፍሮ) , አልበም (ቴዲ_አፍሮ , B) , አልበም_ስም (B , ያስተሰርያል) , አርዕስት (B , A))	count (count (A) , count (B))	0.15	0.14	0.11	0.10
48	የትግራይ ክልል ዋና ከተማ የህዝብ ቁጥር ስንት ነው	answer (A , ዋና_ከተማ (ትግራይ_ክልል , B) , ህዝብ_ብዛት (B , A))	answer (A , ዋና_ከተማ (ትግራይ_ክልል , B) , ህዝብ_ብዛት (B , A))	1.0	1.0	1.0	1.0
49	የአስቴር አወቀ ጨጨሆ የሚለው የአልበም ስም መቸ ተለቀቀ	answer (A , ዘፋኝ (አስቴር_አወቀ) , አልበም (አስቴር_አወቀ , B) , አልበም_ስም (B , ጨጨሆ) , የተለቀቀበት_ቀን (B , A))	A , (count (A) , count (A)) , count (A)) , count (	0.19	0.12	0.10	0.11
50	የእጅጋየሁ ሽባባው ጂጂ የሚባለው አልበም በውስጡ የያዛቸውን አርስቶች ጥቀስ	answer (A , ዘፋኝ (እጅጋየሁ_ሽባባው) , አልበም (እጅጋየሁ_ሽባባው , B) , አልበም_ስም (B , ጂጂ) , አርዕስት (B , A))	count (A) , count (B)))	0.15	0.12	0.14	0.15

Annex D: All relevance merics for each test question of NSPM with Stemmed

This table shows recall, precision and f-measure values for each processed question. Questions printed in cells with red background were not parsed, questions in white cells succeeded. Questions in yellow cell fail due to an inference problem.

Qu ID	Question	Recall	Precision	F-measure
1	በሀረሪ ክልል ውስጥ የሚገኙትን ሀይቆች ስንት ናቸው	1.0	0.8	0.89
2	ኦሮሚያ ክልል ውስጥ ሸናሻ ይነገራል	1.0	1.0	1.0
3	አዳማ የሚገኘው ስታድየም የተገነባው መቸ ነው	0.0	0.0	0.0
4	አዲስ አበባ የሚገኙት ስታድየም ስንት ናቸው	1.0	0.50	0.67
5	ማሞ ወልዴ ባጠቃላይ ስንት ብር አስገኝቷል	1.0	1.0	1.0
6	የኢትዮጵያ የሴቶች ብሄራዊ ቡድን መሀል አጥቂ ስንት ናቸው	0.0	0.0	0.0
7	በቤንሻንጉል ጉሙዝ ክልል የሚገኙትን ሀይቆች ጥቀስ	1.0	0.8	0.89
8	ጉና ተራራ የሚገኘው ሀረሪ ክልል ነው	1.0	1.0	1.0
9	ቢንያም ደምጤ የሚጫወተው መሀል አጥቂ ቦታ ላይ ነው	1.0	1.0	1.0
10	በኢትዮጵያ የሚገኘው ረዝሙ ተራራ ማነው	0.0	0.0	1.0
11	የእጅጋዬሁ ዲባባ እህቶች ባጠቃላይ ስንት ብር አስመዝገቡ	0.0	0.0	0.0
12	ጥሩነሽ ዲባባ ከሴት እሯጮች በወርቅ ብዛት ትበልጣለች	0.0	0.0	0.0
13	አበበ ቢቂላ የ5000 ሜትር እሯጭ ነው	1.0	1.0	1.0
14	እጅጋዬሁ ዲባባ ስንት እሯጭ እህቶች አሏት	0.0	0.0	0.0
15	መሰረት ደፋር ምን ምን እሩጫ ላይ ተሳተፋለች	1.0	0.50	0.67
16	የፊት መስመር እግርኳስ ተጨዋቾችን ጥቀስልኝ	1.0	1.0	1.0
17	ከደራርቱ ቱሉ እና ከገንዘቤ ዲባባ በሜዳሊያ ብዛት ማን ያንሳል	0.0	0.0	0.0
18	ደደቢት እግርኳስ ክለብ ዋንጫ በልቶ ያውቃል	0.0	0.0	0.0
19	ቀነኒሳ በቀለ ስንት የሩጫ አይቶች ላይ ተሳተፈ	1.0	1.0	1.0
20	አዳማ የሀረሪ ክልል ዋና ከተማ ነው	1.0	0.8	0.89
21	የሲዳማ ቡና ቅጽል ስም ማን ይባላል	1.0	0.33	0.50
22	አባይ ሀይቅ በስፋት ከጣና ሀይቅ ያንሳል	0.0	0.0	0.0
23	በቤንሻንጉል ጉሙዝ ክልል የሚገኙትን ታሪካዊ ቦታውችን ዘርዝር	1.0	0.50	0.67
24	ሶማሌ ክልል ውስጥ የሚነገሩ ቋንቋዎችን ስንት ናቸው	1.0	1.0	1.0

25	ኡመድ ኡክሪ የተጫወተባቸው ክለቦችን ጥቀስ	1.0	0.50	0.89
26	ዘንገና ሀይቅ ንጹህ ውሀ ሀይቅ ነው	1.0	1.0	1.0
27	በርታ የት ክልል ነው የሚነገረው	1.0	0.8	0.89
28	ጭላሎ ተራራ የሚገኘው ቤንሻንጉል ጉሙዝ ክልል ነው	1.0	0.5	0.67
29	ኦሮሚያ ክልል የሚገኙትን ሀይቅ ጥቀስ	1.0	0.50	0.89
30	በሀረሪ ክልል ከሚገኙት ተራሮች መካከል በርዝመት ትንሹ ተራራ ከፍታው ስንት ነው	0.0	0.0	0.0
31	ከአፋር ክልል እና ከትግራይ ክልል በተራራ ብዛት ማን ይበልጣል	1.0	0.33	0.50
32	የጋምቤላ ክልል ተወላጅ የሆኑትን እጽፈጮችን ዘርዝር	1.0	0.8	0.89
33	የአፋር ክልል በህዝብ ብዛት ከትግራይ ክልል ይበልጣል	0.0	0.0	0.0
34	የትኛው የክልል ዋና ከተማ ነው ትንሽ የህዝብ ቁጥር ያለው	1.0	0.50	0.67
35	ትግራይ ስታድየም ተመልካች በመያዝ አቅም ከአድስ አበባ ስታድየም ይበልጣል	0.0	0.0	0.0
36	የፋሲል ከነማ ስፖርት ክለብ መቾ ተመሰረተ	1.0	0.5	0.67
37	በርዝመት ትልቁ ተራራ የት ክልል ውስጥ ይገኛል	0.0	0.0	0.0
38	ሀይሌ ገብረስላሴ የት ነው የተወለደው	1.0	0.8	0.89
39	ብዛት ዋንጫ የበላው ፉትቦል ቡድን ማነው	0.0	0.0	0.0
40	ደራርቱ ቱሉ ያገኘችውን ሽልማቶች ተናገር	1.0	0.33	0.50
41	ለኢትዮጵያ ብሄራዊ ቡድን ብዙ ጎል ያስገባው ተጨዋች ማነው	0.0	0.0	0.0
42	ሲሳይ አብርሀም የማን እግርኳስ ክለብ ሊቀመንበር ነው	1.0	0.5	0.67
43	ከወንድ እጽፈጮች መካከል ብዙ ሜዳልያ ያስመዘገበው ማነው	0.0	0.0	0.0
44	1500 ሜትር እሩጫ ላይ የተሳተፉ የኢትዮጵያ እጽፈጮችን ዘርዝር	1.0	0.5	0.67
45	በኦሮሚያ ክልል የሚገኘው በረዝመት ትንሹ ተራራ ማነው	0.0	0.0	0.0
46	ታሪኩ በቀለ የት ነው የተወለደው	1.0	1.0	1.0
47	የቴዲ አፍሮ ያስተሰርያል የሚለው የአልበም ስም በውስጡ ስንት ዘፈኖች አሉት	0.0	0.0	0.0
48	የትግራይ ክልል ዋና ከተማ የህዝብ ቁጥር ስንት ነው	1.0	0.5	0.67
49	የአስቴር አወቀ ጨጨሆ የሚለው የአልበም ስም መቾ ተለቀቀ	0.0	0.0	0.0
50	የእጅጋየሁ ሽባባው ጂጂ የሚባለው አልበም በውስጡ የያዛቸውን አርስቶች ጥቀስ	0.0	0.0	0.0

Annex E: Expert Review Questionnaire to Validate Test Data Benchmark for Semantic Question Answering over Amharic Linked Data

Core Requirement	Verification criteria	Strongly agree	Agree	Undecided	Disagree	Strongly Disagree	No Idea
		5	4	3	2	1	0
Comprehensive question set (i.e. Maps to hypotheses context: expressive queries over open domain dataset.)	The question set contains instance reference						
	The question set contains class reference						
	The question set contains operator reference						
	The query set provide a comprehensive combination of different query patterns						
	The set of unique triple patterns are evenly distributed						
	complex queries have fair distribution as queries asking for an instance and an attribute						
	The question set contains INSTANCE-PREDICATE-VARIABLE triple pattern reference						
	The question set contains						

	VARIABLE-PREDICATE-VARIABLE triple pattern reference							
	The question set contains VARIABLE-TYPE-CLASS triple pattern reference							
	The question set contains VARIABLE-PREDICATE-VALUE triple pattern reference							
	The question set contains ORDER triple pattern reference							
	The question set contains AGGREGATE triple pattern reference							
	The question set contains COMPARISON triple pattern reference							
	The set of different question types are fairly distributed (fact/list, yes/no, aggregation, comparison, superlative , and questions require fused information across different dataset sources)							
Semantic formalism	Question ID **#1**	The corresponding logic form maps to question convinced						

(**hints: the other question IDs have same format as Question ID #1**)		in terms of: **entity/variable**						
		The corresponding logic form maps to question convinced in terms of: **relation/predicate**						
		The corresponding logic form maps to question convinced in terms of: **operator**						
	Question ID **#2**	entity/variable						
		relation/predicate						
		operator						
	Question ID **#3**	entity/variable						
		relation/predicate						
		operator						
	. . .							
	Question ID **#50**	entity/variable						
		relation/predicate						
		operator						

Annex F: Part of Amahric Query Set

The list below provides the set of natural language queries and the corresponding SPARQL queries and logic forms for the Amharic Semantic Query Answering over Linked Data test collection.

<question id="1" answertype="number" aggregation="true" order="false" fused="false" comparison="false">
<string> በሀረሪ ክልል ውስጥ የሚገኙትን ሀይቆች ስንት ናቸው</string>
<lf> answer (count (A) , ሀይቅ (A), መገኛ_ክልል (A , ሀረሪ_ክልል)) </lf>
<query>
SELECT COUNT(?uri) WHERE {
?uri <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://am.geo.org/resource/ሀይቅ>.
?uri <http://am.geo.org/property/ክልል> <http://am.geo.org/resource/ሀረሪ>.
}
</query>
<answers>
<answer>
<number>0</number>
</answer>
</answers>
</question>
<question id="2" answertype="boolean" aggregation="false" order="false" fused="false" comparison="false">
<string> ኦሮሚያ ክልል ውስጥ ሸናሻ ይነገራል?</string>
<lf> answer (? , የሚነገር_ቋንቋ (ኦሮሚያ_ክልል , ሸናሻ)) </lf>
<query>
ASK
WHERE {
<http://am.geo.org/resource/ኦሮሚያ> <http://am.geo.org/property/የሚነገሩ_ቋንቋ> "ሸናሻ"@am.
}
</query>
<answers>
<answer>
<string>false</string>
</answer>
</answers>
</question>
<question id="3" answertype="literal" aggregation="false" order="false" fused="false" comparison="false">
<string> አዳማ የሚገኘው ስታድየም የተገነባው መቼ ነው? </string>
<lf> answer (A , ስታድየም (B) , መገኛ_ቦታ (B , አዳማ) , ግንባታ_ቀን (B , A)) </lf>
<query>
SELECT ?uri WHERE {
?A <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://am.geo.org/resource/ስታድየም>
?A <http://am.sport.org/property/የሚገኝበት_ቦታ> <http://am.geo.org/resource/አዳማ>.
?A <http://am.sport.org/property/የተገነባበት_ቀን> ?uri.
}
</query>
<answers>
<answer>
<string>1972 አም</string>
</answer>
</answers>
</question>
<question id="4" answertype="number" aggregation="true" order="false" fused="false" comparison="false">
<string> አዲስ አበባ የሚገኙት ስታድየም ስንት ናቸው?</string>
<lf> answer (count (A) , ስታድየም (B) , መገኛ_ቦታ (A , አዲስ_አበባ)) </lf>

```
  <query>
    SELECT COUNT(?uri) WHERE {
      ?uri <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://am.geo.org/resource/ስታድየም>
      ?uri <http://am.sport.org/property/የሚገኝበት_ቦታ> <http://am.dbpedia.org/resource/አዲስ_አበባ>.
    }
  </query>
  <answers>
    <answer>
      <number> 2 </number>
    </answer>
  </answers>
</question>
<question id="5" answertype="number" aggregation="false" order="false" fused="false" comparison="false">
  <string> ማሞ ወልዴ ባጠቃላይ ስንት ብር አሸንፏል?</string>
  <lf> answer ( A , የተመዘገበ_ብር ( ማሞ_ወልዴ , A ) ) </lf>
  <query>
    SELECT ?uri WHERE {
      <http://am.sport.org/resource/ማሞ_ወልዴ> <http://am.sport.org/property/ብር_አጠቃላይ> ?uri.
    }
  </query>
  <answers>
    <answer>
      <number>ራስ ዳሽን </number>
    </answer>
  </answers>
</question>
<question id="6" answertype="number" aggregation="true" order="false" fused="false" comparison="false">
  <string> የኢትዮጵያ የሴቶች ብሄራዊ ቡድን መሀል አጥቂ ስንት ናቸው?</string>
  <lf> answer ( count ( A ) , ተጨዋች ( ኢትዮጵያ_ሴቶች_ብሄራዊ_ቡድን , A ) , ጨዋታ_ቦታ ( A , መሀል_አጥቂ ) ) </lf>
  <query>
    SELECT count(?uri) WHERE {
      <http://am.sport.org/resource/ኢትዮጵያ_የሴቶች_ብሄራዊ_እግርኳስ_ቡድን> <http://am.sport.org/property/ተጨዋች>
?uri.
      ?uri <http://am.sport.org/property/የሚጫወትበት_ቦታ> "የመሀል አጥቂ"@am.
    }
  </query>
  <answers>
    <answer>
      <number> 3 </number>
    </answer>
  </answers>
</question>
<question id="7" answertype="resource" aggregation="false" order="false" fused="false" comparison="false">
  <string> በቤንሻንጉል ጉሙዝ ክልል የሚገኙትን ሀይቆች ጥቀስ</string>
  <lf> answer ( A , ሀይቅ ( A ), መኖ_ክልል ( A , ቤንሻንጉል_ጉሙዝ_ክልል ) ) </lf>
  <query>
    SELECT ?uri WHERE {
      ?uri <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://am.geo.org/resource/ሀይቅ>.
      ?uri  <http://am.geo.org/property/ክልል> <http://am.geo.org/resource/ቤንሻንጉል_ጉሙዝ>.
    }
  </query>
  <answers>
    <answer>
      <uri>None</uri>
      <string>None</string>
    </answer>
  </answers>
</question>
```

<question id="8" answertype="boolean" aggregation="false" order="false" fused="false" comparison="false">
<string> ጉና ተራራ የሚገኘው ሀረሪ ክልል ነው?</string>
<lf> answer (? , መገኛ_ክልል (ጉና_ተራራ , ሀረሪ_ክልል)) </lf>
<query>
ASK
WHERE {
<http://am.geo.org/resource/ጉና_ተራራ> <http://am.geo.org/property/ክልል> <http://am.geo.org/resource/ሀረሪ>.
}
</query>
<answers>
<answer>
<string>false </string>
</answer>
</answers>
</question>
<question id="9" answertype="boolean" aggregation="false" order="false" fused="false" comparison="false">
<string> ቢንያም ደምጤ የሚጫወተው መሀል አጥቂ ቦታ ላይ ነው?</string>
<lf> answer (? , እግርኳስ_ተጨዋች (ቢንያም_ደምጤ) , ጨዋታ_ቦታ (ቢንያም_ደምጤ , መሀል_አጥቂ)) </lf>
<query>
ASK
WHERE {
<http://am.sport.org/resource/ቢንያም_ደምጤ> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://am.sport.org/resource/እግርኮስ_ተጨዋች>.
<http://am.sport.org/resource/ቢንያም_ደምጤ> <http://am.sport.org/property/የሚጫወትበት_ቦታ> "የመሀል
አጥቂ"@am .
}
</query>
<answers>
<answer>
<string> true </string>
</answer>
</answers>
</question>
<question id="10" answertype="resource" aggregation="false" order="true" fused="false" comparison="false">
<string> በኢትዮጵያ የሚገኘው ረዘሙ ተራራ ማነው?</string>
<lf> answer (A , argmax ((ተራራ (A), መገኛ_ሀገር (A , ኢትዮጵያ) , ከፍታ (A , B)) , B)) </lf>
<query>
SELECT ?uri WHERE {
?uri rdf:type <http://am.geo.org/resource/ተራራ>.
?uri <http://am.geo.org/property/ክልል> ?o.
?o <http://am.geo.org/property/ሀገር> <http://am.dbpedia.org/resource/ኢትዮጵያ>.
?uri <http://am.dbpedia.org/property/ከፍታ> ?B.
}
order by desc(?B) limit 1 offset 0
</query>
<answers>
<answer>
<uri> http://am.geo.org/resource/ራስዳሽን_ተራራ </uri>
<string>ራስ ዳሽን </string>
</answer>
</answers>
</question>
<question id="11" answertype="resource" aggregation="false" order="true" fused="false" comparison="false">
<string> የእጅጋዬሁ ዲባባ እህቶች ባጠቃላይ ሰንት ብር አስመዝግቡ?</string>
<lf> answer (count (A) , የተመዘገቢ_ብር (B , A) , እህት (B , እጅጋዬሁ_ዲባባ)) </lf>
<query>
SELECT ?uri WHERE {
?A <http://am.sport.org/property/እህት> <http://am.sport.org/resource/እጅጋየሁ_ዲባባ> ..

```
        ?A <http://am.sport.org/property/ብር_አጠቃላይ> ?uri.
      }
  </query>
  <answers>
    <answer>
      <string> 2 </string>
    </answer>
    <answer>
      <string> 4 </string>
    </answer>
  </answers>
</question>
<question id="12" answertype="boolean" aggregation="false" order="false" fused="false" comparison="true">
  <string> ጥሩነሽ ዲባባ ከሴት አሯጮች በወርቅ ብዛት ትበልጣለች?</string>
  <lf> answer ( ? , የተመዘገቢ_ወርቅ ( ጥሩነሽ_ዲባባ , B ) , የተመዘገቢ_ወርቅ ( A , C ) , አሯጭ ( A ) , ይበልጣል ( B , C ) )
</lf>
  <query>
      ASK
      WHERE {
          <http://am.sport.org/resource/ጥሩነሽ_ዲባባ> <http://am.sport.org/property/ወርቅ_አጠቃላይ> ?B.
          ?A  <http://am.sport.org/property/ወርቅ_አጠቃላይ> ?C.
          FILTER (?B > ?C)
      }
  </query>
  <answers>
    <answer>
      <string> true </string>
    </answer>
  </answers>
</question>
<question id="13" answertype="boolean" aggregation="false" order="false" fused="false" comparison="false">
  <string> አበበ ቢቂላ የ5000 ሜትር አሯጭ ነው?</string>
  <lf> answer ( ? , የተሳተፈበት_ሩጫ_አይነት ( አበበ_ቢቂላ , 5000_ሜትር ) , አሯጭ ( አበበ_ቢቂላ ) ) </lf>
  <query>
      ASK
      WHERE {
          <http://am.sport.org/resource/አበበ_ቢቂላ> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://am.sport.org/resource/አሯጭ>.
          <http://am.sport.org/resource/አበበ_ቢቂላ> <http://am.sport.org/property/የተሳተፈው_እሩጫ> "5000 ሜትር"@am.
      }
  </query>
  <answers>
    <answer>
      <string> true </string>
    </answer>
  </answers>
</question>
<question id="14" answertype="number" aggregation="false" order="false" fused="false" comparison="false">
  <string> እጅጋዬሁ ዲባባ ሰንት አሯጭ እህቶች አሏት?</string>
  <lf> answer ( count ( A ) , እህት ( A , እጅጋዬሁ_ዲባባ ) , አሯጭ ( A ) ) </lf>
  <query>
      SELECT COUNT(?uri) WHERE {
          ?uri  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://am.sport.org/resource/እሩጫ>.
          ?uri <http://am.sport.org/property/እህት> <http://am.sport.org/resource/እጅጋዬሁ_ዲባባ>.
      }
  </query>
  <answers>
    <answer>
```

<number> 2 </number>
</answer>
</answers>
</question>
<question id="15" answertype="literal" aggregation="false" order="false" fused="false" comparison="false">
<string> መሰረት ደፋር ምን ምን እሩጫ ላይ ተሳተፋለች?</string>
<lf> answer (A , የተሳተፈበት_ሩጫ_አይነት (መሰረት_ደፋር , A)) </lf>
<query>
SELECT ?uri WHERE {
<http://am.sport.org/resource/መሰረት_ደፋር> <http://am.sport.org/property/የተሳተፈው_እሩጫ> ?uri.
}
</query>
<answers>
<answer>
<string>የረዝም እርቀት እሩጫ </string>
</answer>
<answer>
<string>3000 ሜትር </string>
</answer>
<answer>
<string>5000 ሜትር </string>
</answer>
<answer>
<string>2 ማይልስ (የቤት ውስጥ) </string>
</answer>
</answers>
</question>
<question id="16" answertype="resource" aggregation="false" order="false" fused="false" comparison="false">
<string> የፊት መሰመር እግርኳስ ተጨዋቾቹን ጥቀሰልኝ</string>
<lf> answer (A , እግርኳስ_ተጨዋች (A) , ጨዋታ_ቦታ (A , አጥቂ)) </lf>
<query>
SELECT ?uri WHERE {
?uri <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://am.sport.org/resource/እግርኮስ_ተጨዋች>.
?uri <http://am.sport.org/property/የሚጫወትበት_ቦታ> "የመሀል አጥቂ"@am .
}
</query>
<answers>
<answer>
<uri> http://am.sport.org/resource/ሳላዲን_ሰይድ </uri>
<string>ሳላዲን ሰይድ </string>
</answer>
<answer>
<uri> http://am.sport.org/resource/ጌታነህ_ከበደ </uri>
<string>ጌታነህ ከበደ </string>
</answer>
<answer>
<uri> http://am.sport.org/resource/ኡመድ_ኡክሪ </uri>
<string>ኡመድ ኡክሪ</string>
</answer>
<answer>
<uri> http://am.sport.org/resource/ዳዊት_ፍቃዱ </uri>
<string>ዳዊት ፍቃዱ </string>
</answer>
<answer>
<uri> http://am.sport.org/resource/ኤፍሬም_አሻሞ</uri>
<string>ኤፍሬም አሻሞ </string>
</answer>
<answer>

<uri> http://am.sport.org/resource/ሽመልስ_በቀለ</uri>
<string>ሽመልስ በቀለ </string>
</answer>
<answer>
<uri> http://am.sport.org/resource/ቢንያም_ደምጤ</uri>
<string>ቢንያም ደምጤ </string>
</answer>
<answer>
<uri> http://am.sport.org/resource/አረሂማን_ቤዛ</uri>
<string>አረሂማን ቤዛ </string>
</answer>
<answer>
<uri> http://am.sport.org/resource/ሽታይ_አባአ</uri>
<string>ሽታይ አባአ </string>
</answer>
<answer>
<uri> http://am.sport.org/resource/ብርቱካን_ዋሬ</uri>
<string>ብርቱካን ዋሬ </string>
</answer>
</answers>
</question>
<question id="17" answertype="resource" aggregation="false" order="false" fused="false" comparison="true">
<string> ***ከደራርቱ ቱሉ እና ከገንዘቤ ዲባባ በሜዳሊያ ብዛት ማን ያንሳል?***</string>
<lf> ***answer (who , ሜዳሊያ (ደራርቱ_ቱሉ , A) , ሜዳሊያ (ገንዘቤ_ዲባባ , B) , ያንሳል (count (A) , count (B)))***
</lf>
<query>
SELECT ?uri WHERE {
<http://am.sport.org/resource/ደራርቱ_ቱሉ> <http://am.sport.org/property/ሜዳሊያዎች> ?A
<http://am.sport.org/resource/ገንዘቤ_ዲባባ> <http://am.sport.org/property/ሜዳሊያዎች> ?B.
bind (if(count(?A)<count(?B),<http://am.sport.org/resource/ደራርቱ_ቱሉ>,<http://am.sport.org/resource/ገንዘቤ_ዲባባ>
) as ?uri)
}
</query>
<answers>
<answer>
<uri> http://am.sport.org/resource/ደራርቱ_ቱሉ </uri>
<string>ደራርቱ ቱሉ </string>
</answer>
</answers>
</question>
<question id="18" answertype="boolean" aggregation="false" order="false" fused="false" comparison="false">
<string> ***ደደቢት እግርኳስ ክለብ ዋንጫ በልቶ ያውቃል?***</string>
<lf> ***answer (? , ዋንጫ_ብዛት (ደደቢት_እግርኳስ_ክለብ , A))*** </lf>
<query>
Ask
WHERE {
<http://am.sport.org/resource/ደደቢት_የእግር_ኳስ_ክለብ> <http://am.sport.org/property/የዋንጫ_ብዛት> ?uri.
FILTER (?uri > 0)
}
</query>
<answers>
<answer>
<string>false </string>
</answer>
</answers>
</question>
<question id="19" answertype="number" aggregation="true" order="false" fused="false" comparison="false">
<string> ***ቀነኒሳ በቀለ ስንት የሩጫ ኢቤቶች ላይ ተሳተፈ?***</string>

<lf> *answer (count (A) , የተሳተፈበት_ሩጫ_አይነት (ቀነኒሳ_በቀለ , A))* </lf>
<query>
SELECT COUNT(?uri) WHERE {
<http://am.sport.org/resource/ቀነኒሳ_በቀለ> <http://am.sport.org/property/የተሳተፈው_እሩጫ> ?uri.
}
</query>
<answers>
<answer>
<number> 4 </number>
</answer>
</answers>
</question>
<**question id="20"** answertype="boolean" aggregation="false" order="false" fused="false" comparison="false">
<string> *አዳማ የሀረሪ ክልል ዋና ከተማ ነው?*</string>
<lf> *answer (? , ዋና_ከተማ (ሀረሪ_ክልል , አዳማ))* </lf>
<query>
ASK
WHERE {
<http://am.geo.org/resource/ሀረሪ> <http://am.geo.org/property/ዋና_ከተማ> <http://am.geo.org/resource/አዳማ>.
}
</query>
<answers>
<answer>
<string>false</string>
</answer>
</answers>
</question>

• • •

<question id="**50**" answertype="literal" aggregation="false" order="false" fused="false" comparison="false">
<string> *የእጅጋየሁ_ሽባባው_ጂጂ የሚባለው አልበም በውስጡ የያዛቸውን አርእስቶች ጥቀስ?*</string>
<lf>*answer (A , ዘፋኝ (እጅጋየሁ_ሽባባው) , አልበም (እጅጋየሁ_ሽባባው , B) , አልበም_ስም (B , ጂጂ) , አርዕስት (B , A)*
)</lf>
<query>
SELECT ?A WHERE {
<http://am.Artists.org/resource/እጅጋየሁ_ሽባባው> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://am.Artists.org/resource/ዘፋኝ>..
<http://am.Artists.org/resource/እጅጋየሁ_ሽባባው> <http://am.Artists.org/property/አልበም> ?uri.
?uri <http://am.dbpedia.org/property/አልበምስም> "ጂጂ"@am.
?uri <http://am.dbpedia.org/property/አርዕስት> ?A.
}
</query>
<answers>
<answer>
<string>ጉድ ፈላ </string>
</answer>
<answer>
<string>መንገደኛ </string>
</answer>
<answer>
<string>ተው አንተ ሰው </string>
</answer>
<answer>
<string>አባይ </string>
</answer>

```
      <answer>
        <string>ባለ ዋሽንቱ </string>
      </answer>
      <answer>
        <string>ጉራማይሌ </string>
      </answer>
      <answer>
        <string>ሰው አርገኝ </string>
      </answer>
      <answer>
        <string>ዓይናማ </string>
      </answer>
      <answer>
        <string>ካህን </string>
      </answer>
      <answer>
        <string>ዘማዬ </string>
      </answer>
      <answer>
        <string>አቤት ውበት </string>
      </answer>
      <answer>
        <string>ናፊቀኝ </string>
      </answer>
      <answer>
        <string>አድዋ </string>
      </answer>
    </answers>
  </question>
```

Signed Declaration Sheet

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

Declared by:

Name: Gashaw Demlew Ayalew

Signature: _____

Date: _____

Confirmed by advisor:

Name: Fekade Getahun (PhD)

Signature: _____

Date: _____