YAŞAR UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

PHD THESIS

DEVELOPMENT OF GEOGRAPHIC TURKISH QUESTION ANSWERING FRAMEWORK OVER LINKED DATA (GEO-TR)

CEREN ÖCAL TAŞAR

THESIS ADVISOR: ASSOC. PROF. DR. MURAT KOMESLİ

COMPUTER ENGINEERING

PRESENTATION DATE: 22.06.2018

BORNOVA / İZMİR
JUNE 2018

We certify that, as the jury, we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of the Doctor of Philosophy.

Jury Members:	**Signature:**
Assoc. Prof. Dr. Murat KOMESLİ Yaşar University	.....
Prof. Dr. Mehmet CUDİ OKUR Yaşar University	.....
Prof. Dr. Vahap TECİM Dokuz Eylül University	.....
Assoc. Prof. Dr. Murat OSMAN ÜNALIR Ege University	.....
Assoc. Prof. Dr. Mehmet Süleyman ÜNLÜTÜRK Yaşar University	.....

Prof. Dr. Cüneyt GÜZELİŞ
Director of the Graduate School

ABSTRACT

DEVELOPMENT OF GEOGRAPHIC TURKISH QUESTION ANSWERING FRAMEWORK OVER LINKED DATA (GEO-TR)

Öcal Taşar, Ceren
PHD, Computer Engineering
Advisor: Assoc. Prof. Dr. Murat KOMESLİ
June 2018

With a considerable growth of linked data, researchers focused on how to increase the availability of the semantic web technologies to provide practical usage for real life systems. Question answering systems are one of these real life systems that communicate directly with the end users, understand user intention and generate answers. End users do not want to care about any structural query language or vocabulary of the knowledge base where the point of the problem arises. In this study, a question-answering framework that converts Turkish natural language input into SPARQL queries in geographic domain is proposed. Additionally, a novel Turkish ontology which covers Chapter 6 of the geography lesson named "Spatial Synthesis: Turkey" in secondary school in 10th grade curriculum, is developed to be used as linked data provider. Later on, a literature gap in Turkish question answering systems, which utilizes linked data in geographical domain, is addressed. Hybrid system architecture that combines natural language processing techniques with linked data technologies to generate answers is also represented. Further related research areas are also suggested.

Key Words: question answering systems, linked data, ontology development, natural language processing

ÖZ

BAĞLI VERİ ÜZERİNDE TÜRKÇE COĞRAFİ SORU CEVAPLAMA ÇERÇEVESİNİN GELİŞTİRİLMESİ (GEO-TR)

Öcal Taşar, Ceren
Doktora Tezi, Bilgisayar Mühendisliği
Danışman: Doç. Dr. Murat KOMESLİ
Haziran 2018

Bağlı verilerdeki kayda değer artış doğrultusunda, araştırmacılar, gerçek hayat sistemlerine pratik kullanım sağlamak için semantik web teknolojilerinin kullanılabilirliğini nasıl arttıracaklarına odaklandılar. Soru cevaplama sistemleri, son kullanıcılarla doğrudan iletişim kuran, kullanıcı isteğini anlayan ve cevaplar üreten bu gerçek hayat sistemlerinden biridir. Son kullanıcılar, herhangi bir yapısal sorgulama dili veya bilgi tabanında geçerli olan kelime bilgisi hakkında bilgi sahibi olmaya ihtiyaç duymak istememektedirler. Bu tez çalışmasında, Türkçe doğal dil soru girdisini SPARQL sorgularına dönüştüren coğrafi bir soru cevaplama çerçevesi sunulmuştur. Bağlı veri sağlayıcı olarak kullanılmak üzere 10. sınıf ortaöğretim müfredatından "Mekansal Sentez: Türkiye" olarak adlandırılan coğrafya dersinin 6. bölümünü kapsayan yeni bir Türkçe ontoloji geliştirilmiştir. Ayrıca, coğrafi alanda bağlantılı verileri kullanan Türkçe soru cevaplama sistemlerinde bir literatür açığı ele alınmıştır. Cevap üretmek için doğal dil işleme teknikleri ile bağlantılı veri teknolojilerini birleştiren hibrit bir sistem mimarisi önerilmiştir. Konu ile ilgili olası ileri seviye araştırmalar önerilmiştir.

Anahtar Kelimeler: soru cevaplama sistemleri, bağlı veri, ontoloji geliştirme, doğal dil işleme

ACKNOWLEDGEMENTS

First of all, I would like to thank my supervisor Assoc. Prof. Dr. Murat Komesli for his great guidance, patience and invaluable support during this study. I am also very thankful to Assoc. Prof. Dr. Murat Osman Ünalır and Prof. Dr. Mehmet Cudi Okur for their great supervision and significant contribution to my thesis. They all believed in me, shared their knowledge and experience with me, and paid attention during all phases of this study. It was and will be a pleasure to work with them.

I am also grateful to Prof. Dr. Bahar Karaoğlan who encouraged me to study PhD in computer engineering and enlightened my way with her great knowledge and support during my graduate study.

My mother and father have been the two most influential people in my life. I wish to thank them for their enduring encouragement, support and love they have given throughout my life. I dedicate this study to my mother Vahide Esin and father Zekai.

Special thanks to my precious husband, Davut Emre Taşar, for all his infinite patience and believing in me. Without his never-ending support and encouragement, it would be very difficult to complete this work.

Ceren Öcal Taşar

İzmir, 2018

TEXT OF OATH

I declare and honestly confirm that my study, titled "DEVELOPMENT OF GEOGRAPHIC TURKISH QUESTION ANSWERING FRAMEWORK OVER LINKED DATA (GEO-TR)" and presented as a PhD Thesis, has been written without applying to any assistance inconsistent with scientific ethics and traditions. I declare, to the best of my knowledge and belief, that all content and ideas drawn directly or indirectly from external sources are indicated in the text and listed in the list of references.

Ceren Öcal Taşar

Signature

.....

July 18, 2018

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SYMBOLS AND ABBREVIATIONS

ABBREVIATIONS:

AI Artificial Intelligence

NER Named Entity Recognition

NL Natural Language

NLI Natural Language Interface

NLP Natural Language Processing

POS Part of Speech Tagging

IR Information Retrieval

SYMBOLS:

q Question input.

CHAPTER 1
INTRODUCTION AND BACKGROUND INFORMATION

Chapter 1 introduces the motivation, scope and the contributions of the study represented in this thesis. Natural language processing, question answering systems and question answering systems that use linked data is discussed to comprehend the major methodologies involved in this thesis. Starting with the explanation of the related concepts helps to better understand the problem and proposed work in Chapter 4. Continuing with the problem definition and motivation better positions this study in the literature. After addressing the gap in the literature, our approach is drawn to fulfill it to indicate contribution of this thesis. Rest of the chapter continues with the sections: natural language processing, question answering systems, question answering systems over linked data, problem definition and motivation, contributions and thesis organization.

1.1. Natural Language Processing

Information extraction, information retrieval and related research areas have faced with new problems as an impact of the massive and rapid increase of data sources. Growing number of users require direct interaction with the computer systems instead of using any formal programming or command languages. To improve the way of interaction between the users and computer systems, researchers are focusing on feasible and effective solutions by converting the information need of user which is expressed in natural language to a machine-readable and machine-understandable format. Natural language (NL) interfaces emerged as a promising way of providing user-system communication. NL interfaces primarily employ natural language processing techniques to morphologically analyze the NL input with the aim of achieving user intent. Firstly, natural language processing (NLP) concept should be defined to better formulate the methodology behind the natural language interfaces.

Natural languages are naturally evolved way of communication between human beings. The term NL stands for differentiating between the languages that human beings use

and formal and structural languages that computers use such as C++, Java or Python. Scientific study of natural languages with computational approach is called natural language processing. Natural language generation and understanding are the major research areas on NLP. For this thesis, natural language understanding is the taken consideration as means to analyze the NL input. To understand natural language from a computational perspective, human language is converted into formal representation such as parse trees or first order logic to provide a machine understandable structure. For a specific human language, structure of the words in other words morphology of the words is modelled in order to understand a sentence or an expression in that language (Kumar, 2011).

As a subfield of Artificial Intelligence (AI) and computational linguistics, primary concern in NLP is processing of the words, sentences and texts to generate the structure of the words and relationships between them. Fundamental principle for NLP can be summarized as enriching the data by applying linguistic methods. Possible number of different words are included in natural language and extending the set of words by using prefixes and suffixes is also possible by making various types of definitions for a given word. This fact makes the set of words in natural language really large and complex. Additionally, considering the features of a natural language, it is not always strictly in accordance with rules of grammar and syntax. All of these factors make NLP a challenging process (Hoque, Rahman and KumarDhar, 2007).

Instead of implementing language-processing tasks with the involvement of direct manual large set of rules formed, recent NLP algorithms are based on mainly statistical machine learning methods. Attempts to use machine learning techniques lower the manual effort and increase the accuracy of the output generated by language processing tasks. Learning approaches provide the inferencing algorithms by generating models to make predictions for the input containing words or structures that have not seen in the training data. General NLP system architecture as a block diagram that accepts input in natural language and generates understood language as output is depicted on Figure 1.

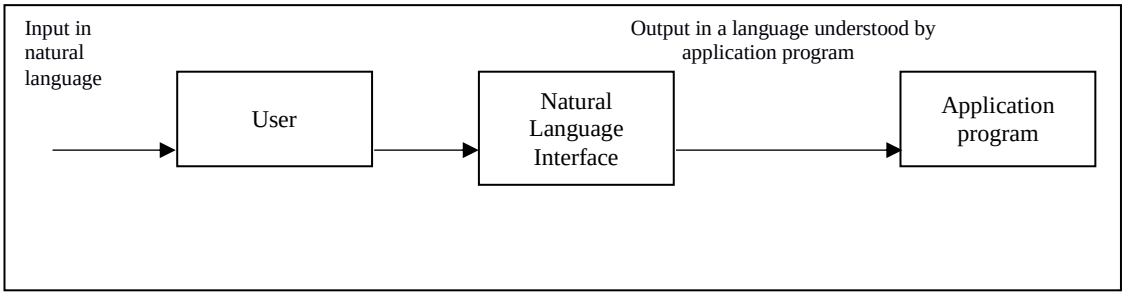


Figure 1.1. Block diagram of a NLP system (Kumar, 2011)

Hayes and Carbonell (1983) defined natural language processing as the formulation and investigation of computationally effective mechanisms for communication through a natural language. Creating models of human language use and making them effectively used by the computer systems are the main motivations of NLP.

Liddy (2001) describes NLP as theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications. Definition of "Theoretically motivated range of computational techniques" expression should be concentrated to gain a clearer perspective on NLP. These computational techniques that form different levels of linguistic analysis that are applied by various NLP applications. Each level handles different types of meanings. For that reason, applications require different levels or combinations of linguistic levels to make their system robust and generate more accurate results.

Sentence or word structures has a key role to extract the meaning based on the analysis of each word for natural language understanding. Sentence understanding consists of two fundamental processes which are morphological and semantic analysis. Here, morphological analysis resides as a prior step while semantic analysis is at higher level (Guo, Li and Shao, 2011). To understand the morphological analysis, at first meaning of the word "morphological" should be concentrated on. Morphological stands for "related to form". So, what is meant by morphological analysis is analyzing only the forms of the words, it is not about meanings (Temizer and Diri, 2012). Morphological structure of the words and the relationships between them are needed to empower semantic analysis. At the syntactic level in morphological analysis, algorithms to predict part-of-speech tags for each word (e.g. noun, verb, and adjective) and

determining the relationships between the words (e.g., subject, object and other modifiers) which is called dependency parsing should be provided. Whereas for the semantic level, noun-phrase extraction (e.g. identifying President of US, CEO in free text), tagging these noun-phrases as either person, organization, location or common noun and resolving mentions of entities in a sentence or free text are performed (Collobert et al., 2011).

Before presenting the question answering systems and problem definition, linguistic analysis techniques applied in this thesis that are part-of-speech tagging, name entity recognition and dependency analysis are briefly described.

Part-of-speech (POS) tagging is an effort to morphological structure of the words (e.g., noun, verb, adjective) in a sentence (Indurkhya and Damerau, 2010). POS tag of a word represents the syntactic role of each word. Systems utilize different algorithms or tools to predict part-of-speech tags for each word at the syntactic analysis level. For further manipulation and subsequent interpretation of the natural language input, identifying the underlying syntactic structure of a sequence of words is assumed as the first step after tokenizing the words of an expression. Best POS classifiers use the machine learning approach based on classifiers trained on windows of text, which are then fed to a bidirectional decoding algorithm during inference (Collobert et al., 2011).

Dependency analysis is one of the primary research topics in natural language processing which performs parsing natural sentences to certain types of structures. Several relationships between words (e.g., subject, object and other modifiers) in a sentence is identified by dependency analysis of a sentence. Relationships between words are represented with dependency structures. Dependency parsing algorithms are based upon dependency nodes and relations. Except for root node, each word-token represented as a node in a dependency structure which is dependent in exactly one other node (Nivre, 2010). It is important to understand the relations between verbs and their arguments to generate dependencies and analyze terms within a sentence. The information about who is doing what to whom defines the partial role of every term in a sentence to achieve whole meaning (Shehata, Karray and Kamel, 2007)

Another important NLP technique is name entity recognition (NER) which is nearly related extracting semantics. Name entity recognizers perform noun-phrase extraction and tagging these noun-phrases as either person, organization, location or common

noun. NER is a critical process of NLP that resolves the mentions in a sentence or free text and is a sub-field of information extraction. NER basically locates and classifies the atomic tokens that represents words into predefined categories. Predefined categories can be customized according to the context and domain of the information need. Context specification plays a critical role to improve accuracy values of natural language understanding. Like in POS tagging, learning methods are also popular while creating the model for name entity recognizer (Celikkaya, Torunoglu, and Eryigit, 2013).

1.2. Question Answering Systems

A very old dream until the time that first computers were invented has always been giving ability of understanding and interpreting human language to computers. Multidisciplinary effort is required that combines speech and language processing, human language technology, natural language processing, computational linguistics, and speech recognition and synthesis. Fundamental target is to improve human-machine and human-human communication and provide useful processing of text or speech. Question answering systems is one of the contributors to this target by producing accurate answers to the questions posed in natural language.

Fundamental tasks for question answering are processing natural language input, understanding user intention and generating accurate results. Commonly used steps used for question answering are question processing, document processing and answer processing. Main approaches described for question answering systems are information retrieval (IR) based question answering and knowledge-based question answering (Jurafsky and Martin, 2014). First approach is also called text-based or document-based question answering focuses information retrieval techniques to generate answers. For the given query in natural language, a search mechanism is employed to produce the candidate documents that may hold the answer to the question. Candidate documents are ranked to extract candidate answer strings from the passages into them. By IR-based question answering, unstructured or semi-structured data is used as data sources. This approach is most appropriate for the systems mostly accept factoid questions. If a question can be answered with simple short expressions and based on a simple fact, then this question falls to the class of factoid questions. “Who invented the iPhone?”, “Where is the capital of Australia?”, “What is the official

language of Turkey?" can be given examples to factoid questions. IR-based is formulated with two main tasks. First task is to decide on answer type and the second task is query formulation. An architecture of a generic IR-based question answering system is illustrated on Figure 2.

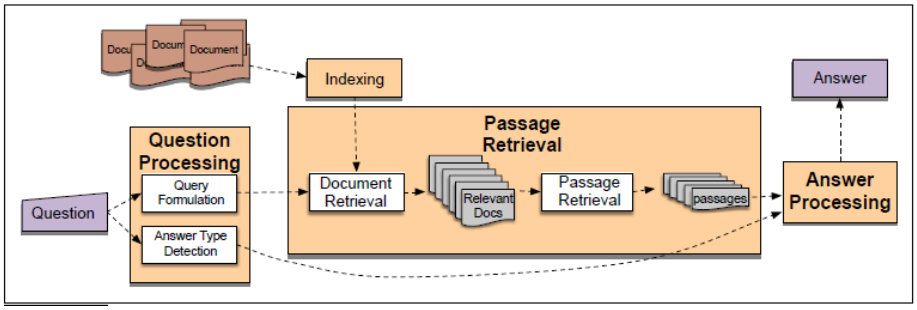


Figure 1.2. A generic IR-based question answering system (Jurafsky and Martin, 2014)

Second approach is knowledge-based question answering which is applied in this thesis study. Knowledge-based question answering methods focus on transforming unstructured natural language to the structured query languages. Systems are employing semantic parsers that converts a text based expression to a logical form. Methods enforced by knowledge-based question answering are rule-based, supervised and semi-supervised. Rule-based methods are based on some rules and patterns to extract relations from the question. Therefore, expert knowledge is required. Supervised methods are using learning algorithms by using annotated train data set that consists of set of questions that are mapped with their logical forms. Generally, supervised learning algorithms utilize the parse trees to perform mapping. For example for a question like "When was Ada Lovelace born?" is mapped with the logical form birth-year (Ada Lovelace, ?x) by using the defined training rule represented in Figure 1.3.

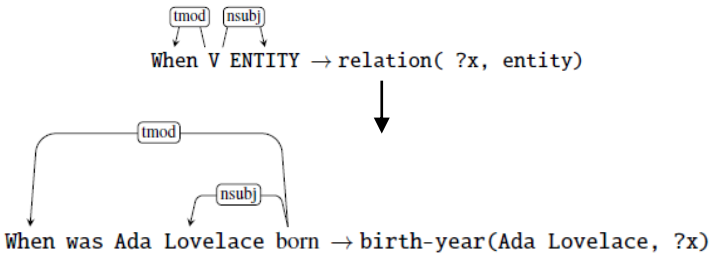


Figure 1.3. Processing of a sample query (Jurafsky and Martin, 2014)

Each row in the training set is parsed to be inferred for a pattern like “When was X born?” is mapped with the relation birth-year in the knowledge base.

More complex questions that holds quantitative reasoning and more than one relation is handled by extending the supervised approach. An example query can be given from GEOQUERY (Zelle and Mooney, 1996) dataset like: “What is the biggest state bordering Texas?” is mapped with the logical form: argmax(λx.state(x)^borders(x, texas), λx.size(x)).

Considerable amount of effort is required to prepare training data sets which results in discovering another approach to provide easier way: semi-supervised methods. Starting with small annotated training data and using supervision distant approaches to widen the coverage to map with logical forms is provided in semi-supervised systems. Rule based and supervised methods are combined in this thesis study.

1.3. Question Answering Over Linked Data

Emerging semantic web technologies enable the growth of linked data and offering a broad spectrum of domain which represents requirements of real life (Berners-Lee, Hendler and Lassila, 2001). As the amount of data is growing in the semantic web, linked data storage which have interlinked RDF datasets is proving source of information, requirement to access and use this source also grows. Following this requirement, researches focus on how end users utilize the knowledge bases in a form of interlinked data without knowing any structured query language like SPARQL (Unger, Freitas and Cimiano, 2014). In spite of the benefits that offered by semantic web technologies, continuous growth of linked data resulted in some challenges and drawbacks in terms of practical usage of it. Understanding the complexity of the rationale behind semantic web is very difficult for end user to practically use it. An obvious gap is seen between users and semantic web technologies. Additionally, searching and sending query to large scale content is really tough to deal with. Considering challenges and drawbacks, researchers who are interested in this field discovered the importance of user friendly interfaces which enable users to search and query this massive content. These interfaces are the major component for question answering frameworks over linked data (Lopez et al., 2011).

IR-based question answering systems offer answer questions in open-domain environments by processing unstructured text and extract the paragraphs or documents

which may hold the answer. Huge amount of data is processed which indicates IR-based approach scale well. However, if the answer is not in the same form like expressed in the query or combination of multiple documents is required to find an answer to the corresponding query, IR-based methods would miss the semantics which is a bottleneck for these methods. Semantic question answering over linked data provides capturing inherited relationships, word meaning links and inferencing techniques (Lopez et al., 2013).

Main concerns for semantic question answering can be listed as: questions are asked in natural language, end users use their own terminology and users achieve an accurate answer by using a RDF knowledge base; ontology. Users do not need to care about the query language and vocabulary of the knowledge base (Höffner, 2017). Natural language is ambiguous and complex. Various steps are applied for question answering over linked data and this steps can be customized according to the needs of applications. A generic question answering framework is illustrated on Figure 4. Generic architecture represented in Figure 4 is customized for this thesis study by combining NLP techniques and linked data technologies which is discussed in Chapter 3.

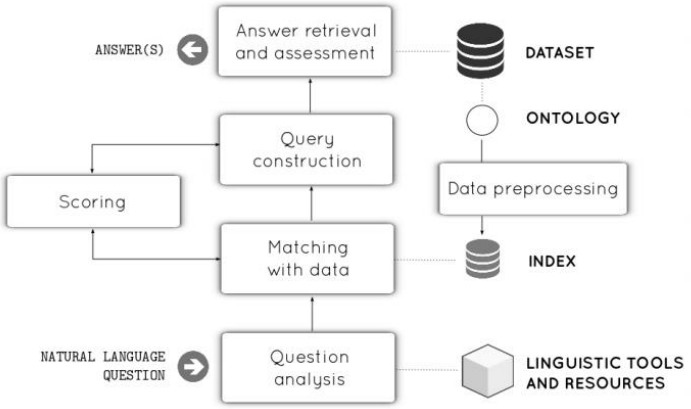


Figure 1.4. High-level components of question answering systems over linked data (Unger, Freitas and Cimiano, 2014)

Fundamental challenge to provide answers over linked data is converting natural language input into SPARQL queries or graphical representations that generate precise and accurate answer when query is executed. Conversion process involves matching query tokens (words) with the represented semantic items (URIs) in the knowledge base. For a query like “When was Abraham Lincoln born?” that uses DBpedia as the knowledge source, the recognized named entity “Abraham Lincoln” in the sentence is mapped to the URI http://dbpedia.org/resource/Abraham Lincoln resource and “born”

needs to be mapped with the relation “birthplace” (URI: http://dbpedia.org/ontology/birthplace). Answer is generated by using corresponding URIs (Walter et al., 2012).

1.4. Problem Definition and Motivation

A literature review study about question answering systems over linked data is performed at the beginning of the thesis study which is mentioned as contributions in the next subsection. Two different types of study is generated during literature review process. First one is a systematic mapping study and the second is a comparative review. By interpreting the outcomes of these studies, it is seen that majority of the studies that provides question answering over linked data exist. However, language used in these systems are other than Turkish which is our own language. Choosing Turkish language for a question answering framework over linked data guarantees the originality and uniqueness and more importantly, considerable contribution is planned for Turkish language in this research field. Then, drawing the boundaries by focusing on a specific domain which is decided as geography. Finally, main aim is set on a Turkish question answering systems for the usage of the students in secondary school in 10th grade for geographical domain and a specific chapter (Chapter 6) which is named “Spatial Synthesis: Turkey” is decided. Final problem definition is how to design and implement a Turkish question answering framework over a geographic ontology which is designed and created in the scope of this thesis study. This ontology is named as GEO-TR which is bounded to answer the questions in the “Spatial Synthesis: Turkey” chapter.

Turkish, which is an agglutinative language and free constituent order language, which makes it morphologically different from the other languages. Additionally, by using same lemma of a word, it is possible to derive many forms that have different meanings. Differences of Turkish are reflected in the use of NLP tools and algorithms designed for generation of SPARQL while designing and implementing pipeline for Turkish question answering over GEO-TR.

In conclusion, main target is the development of Turkish question answering framework over GEO-TR which is an ontology developed by following the procedure ontology development 101 (Noy and McGuinness, 2001) under the context of this thesis study. Fulfilling the gap between students as end users and ontologies and giving them an ability to ask questions in natural language without knowing any other

structural query language are all motivated this thesis study. In addition to this, contributing to Turkish question answering systems over linked data is another motivation of this thesis study.

1.5. Contributions

This thesis study is initiated with surveying the literature for question answering frameworks over linked data. After performing a survey, an idea came out about requirement of a systematic method that would ease the way of review, perform summarization and combines required information about these studies systematically. Therefore, two main contributions are motivated during this thesis study.

1) First contribution is a systematic mapping study on question answering framework over linked data. Studies published between the years 2010 and 2017 from major databases, journals and proceedings of conferences or workshops are identified and analyzed to conduct a systematic mapping study. Research scope is determined by identifying the research question and inclusion/exclusion criteria. By using the defined steps for study selection phase, 53 primary studies are selected from an initial set of 845 studies. Methods employed for question answering over linked data, gaps between the required and current approaches and popular approaches which has recently emerged are discussed. Finally, a comparison is made between the represented study and related work to declare the different points and contributions of the represented study. This study is accepted in IET Software journal.

2) Second contribution is a comparative review for question answering frameworks on the linked data. State-of-art question answering systems that uses semantic endpoints are examined in detail. Systematic literature review is applied as research method for this review study. Systems that combine natural language processing techniques and linked data technologies are in the focus set of this study. Research scope is determined by identifying the research question and inclusion/exclusion criteria. 9 studies remained after gradual selection of studies. Remained studies are explained and compared between each other in terms of precision, recall and F-measure. Fundamental aim is to figure out the points while developing a question answering systems that accept input in natural language and converting it into SPARQL. Finally, results of the comparison and future research suggestions are given. This study is under review in Romanian Journal of Information Science and Technology.

1.6. Thesis Organization

Thesis study is designed to start with an introduction in the current chapter. Chapter 2 continues with summarization of state of art techniques; a literature review is represented by applying a systematic procedure. Additionally, a comparison is made between existing studies. Chapter 3 introduces geographic Turkish question answering framework over linked data and ontology which is named GEO-TR in the context of this thesis study. Details about question answering methods, steps involved in the pipeline of the framework and algorithms proposed are defined. Combination of NLP-based and ontology-based approach is represented as a hybrid methodology. A comparison is made between hybrid approach and ontology-based approach to better formulate the details of these methodologies. In Chapter 4, discussions about the compared approaches represented in previous chapter and extracted results are given. Finally, future directions and conclusions are drawn.

CHAPTER 2
LITERATURE REVIEW

2.1. Introduction

Large and continuously increasing volume of data resources results in the discussion of new problems in information extraction/retrieval and related research areas. Increasing demand is seen on the request of users which desire to communicate with computer systems and applications in an informal way by learning and using scripting languages. Users do not want use any formal or structural language to communicate with computers and mobile devices. Researchers who discover this requirement and interested in the mentioned research fields, are focusing how to represent and convert user intention in a structural and machine understandable knowledge representation. A combined solution that integrates natural language processing techniques with semantic technologies provides the combination of syntactic and semantic analysis results. Not only analyzing the sentence by using NLP methods but also achieving semantics of the sentence is required for a feasible and effective solution.

Definition of NLP by Hayes and Carbonell (1983) is the formulation and investigation of computationally effective mechanisms for communication through a natural language. NLP is the sub-field of artificial intelligence (AI) and computational linguistics. Computational linguistics focuses on sentence, word and text pre-processing to find the entities and relations/dependencies between words. Artificial intelligence approach can be used to improve the methodologies in computational linguistics. Enrichment of data in natural language format is provided by NLP with applying linguistic methods. By using prefixes and suffixes, a word can have many derivations which makes word set of a language very complex and huge. Especially for Turkish which is an agglutinative language, meaning of the word is changed while the adding prefixes or suffixes are added. Another challenging feature to deal with for NLP systems or applications is that grammar and syntactic rules are not always taken consideration during communication while using natural language. Features mentioned that forms the inherent of NLP make NLP challenging and ambiguous process (Hoque, Rahman and KumarDhar, 2007). Question answering systems that communicates target users with natural language and focus on user intention have all these challenges and techniques to enrich the input to improve software quality

attributes and quality metrics for question answering systems (precision, recall, F-measure etc.) are employed.

Fundamental component behind semantic technologies is the ontologies which defines conceptualization by defining a taxonomy and relations between the concepts along a specific domain. In other words, they describe semantic network of concepts and relations. Notable amount of structural data storage is provided by ontologies. Ontologies are playing crucial role to share and exchange data between different platforms. Therefore, a research trend has grown up about ontology concept, ontology syntax and representation like RDF, OWL and query languages for ontology like RDQL, SPARQL (Berners-Lee, Hendler and Lassila, 2001). For question answering frameworks, NLP techniques has gained new aspect as semantic technologies spread over increasingly. Hybrid solutions that combines NLP techniques with semantic technologies to understand user intention and produce answers to their questions are started to be discussed. First issue is about how semantic technologies help to empower the NLP methods. In other words, how to enrich the NLP results for knowledge extraction and word sense disambiguation. Another point is how NLP helps to manage semantic web development in ontology learning, ontology query and multilingual ontology mapping (Guo and Ren, 2009).

Specification and representation of meaning in computational world is a challenging constraint while understanding an expression. This constraint can be handled with by integrating words into a meaning for the parsed string of words (Pugsee, Evens and Rivepiboon, 2006). Important aspect to be focused on is defining the relations between words, syntactic representation for each word for further analysis and interpretation to achieve the meaning of an expression (Indurkhya and Damerau, 2010). In addition to this, defining the relations between related arguments of verbs to achieve dependencies and "who is doing what to whom" information plays a critical role to determine the whole meaning (Shehata, Karray and Kamel, 2007).

Morphological analysis result of each word in a sentence has great contribution when sentence level natural language understanding is considered. Phases for sentence understanding process has classified into two fundamental parts which are analyzing the sentence morphologically and semantically. Considering a comparison between morphological and semantic analysis, morphological analysis locates in a lower level whereas semantic analysis has location in higher points. Height represents degree of

the closeness to meaning (Guo, Li and Shao, 2011). To understand the difference between them, firstly definitions for both terms should be formulated. Meaning of the word “morphological” is related to form so morphological analysis refers to analyze only the forms of the words. Any determining and analyzing the words semantically is not provided. Morphological analysis resides in the base level and semantics of an expression, word or sentence is not involved (Temizer and Diri, 2012). While going one-step upward to achieve semantics, morphological analysis acts like a ladder that provides morphological structure of the words and relations between them. Part-of-speech (POS) tag (noun, verb, adjective, etc.) prediction for each word and dependency parsing which is responsible for finding relations between words like (subject, object and other modifiers) is served at the syntactic level of morphological analysis. Whereas for semantic level, named entity recognition which performs tagging expressions as either person, organization or location and resolving mentions of entities in a text or sentence are accomplished (Collobert et al., 2011).

Combining ontologies to the hybrid approach assures further enrichment of data. Especially for question answering systems, ontologies can be used as source of information which will result in the requirement of converting natural language input to ontology query languages. Without having knowledge of any formal structural query language, schema or vocabulary, users can utilize from ontologies (Bernstein et al., 2005). A natural language interface that bridges the gap between real-world users and linked data technologies is the focused aspect for this literature review.

Systems that combine natural language techniques and linked data technologies for question answering frameworks are in the domain set of this literature review. To summarize the state of art studies, a systematic research method that identifies and evaluates similar studies is conducted. Systematic research method is applied to review the literature in a structural and efficient way. Cutting edge studies which built in similar architectures with this thesis study are selected according to the common data items specified in the research method. Literature review section is structured as follows: after introducing the research method, scope, directives and research questions, data items and evaluation paradigms are defined. Next part continues with describing the studies selection process and evaluation of the studies due to data items. Next, outcomes are given by comparing the selected studies and literature gap is

extracted from the results. Finally, differences and contributions of our study are pointed out.

2.2. Research Method

Methods to conduct systematic literature review are selected as research method of the literature review section. Systematic literature review is a study which aims to identify, evaluate and comment on empirical studies that all are relevant to a specific research area, question, reason or phenomenon of interest (Kitchenham and Charters, 2007). Evidence based paradigm is applied to generate an objective summary of the relevant studies (Brereton et al., 2007). The idea that supports the requirement of discovering common practices and synthesizing results of the common objectives is used in evidence-based paradigm to produce a professional practice as an outcome of the study (Sackett, 1997).

Literature review section of this study uses the principles of conducting systematic literature review (Kitchenham and Charters, 2007; Brereton et al., 2007). Sequential steps of the research method that produced from that combination that are depicted on Figure 1:

1) Identify research questions
2) Identify research scope
3) Define data items
4) Select and evaluate studies
5) Compare and assess studies
6) Synthesize data

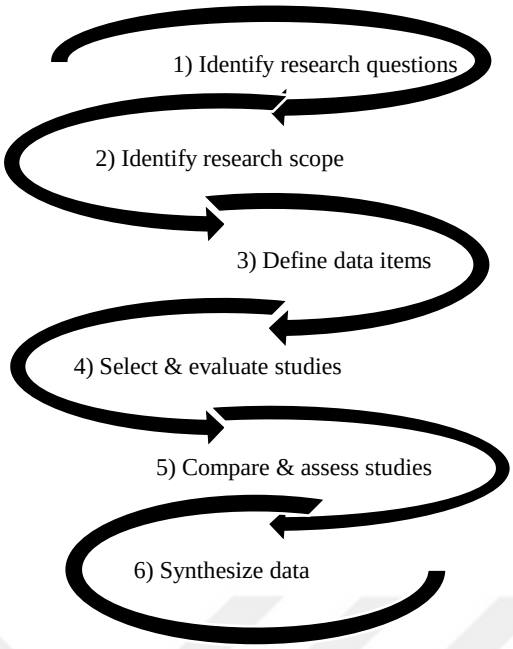


Figure 2.1. Steps of Literature Review

2.2.1. Research Questions

Literature review section is used to enlighten the methodologies and targets of this thesis study about how to combine NLP techniques with linked data technologies in an efficient way. Another important point is to examine the approaches and architectures (sometimes technology decision) they proposed to make unique contribution to literature in an original way. Some boundaries needs to be drawn to illustrate the main issues for literature review. Research questions are shaped by system objective and context of thesis study. The following research questions described below are focused for literature review:

-Q1: What are the cutting-edge techniques to combine syntactic and semantic analysis to enrich natural language input for question answering frameworks?

-Q2: What tools and methods used for to implement these frameworks mentioned in Q1?

-Q3: Which factors should be considered when using question answering frameworks to improve the response accuracy rate?

Research questions are employed to examine similar studies that are published on this topic by using various methods or tools. Neither using ontologies directly by using query languages nor only analyzing the sentence syntactically with NLP based

approaches is the common issue in these studies. Similar to our approach, each of them is focusing on providing unstructured resource enrichment for question answering frameworks by combining NLP and ontology based approaches. Converting natural language queries into ontology query language SPARQL is the common aim of these relevant studies. In other words, unstructured natural language expression that represents user intention is converted to a structural linked data query.

Requirement of discussion and examination on methods or tools that are utilized for similar studies arise to position our study in a unique location. Methods or tools both for NLP and ontology based approaches are taken consideration to answer the research questions. After determining the research questions, boundary lines of the literature review is more precise to define research scope.

2.2.2. Research Scope

In the direction of research questions, selection of similar studies and research scope identification is simplified. While specifying research scope, data items which represents research variables to systematically locate and control the flow of the literature review are extracted. By eliminating the search results due to data items and inclusion/exclusion criteria which will be specified later in Section 2.4.1, final set of similar studies are remained. Articles from several journals are combined and especially, QALD-4 (Unger et al., 2014) and QALD-5 (Unger et al., 2015) environment belong to series of evaluation campaigns on multilingual question answering over linked data are concentrated on. Diversified tasks of QALD-4 and 5 like multilingual question answering over linked data, biomedical question answering over interlinked data and hybrid question answering are combined according to the data items extracted for research scope. Methodology is broaden by applying the empirical studies after ensuring research scope is designated precisely.

2.2.3. Data Items and Evaluation Paradigms

Data items are research paradigms which are mentioned in relevant studies but different methodologies, architectures or tools applied for them. In order to discuss from an information retrieval perspective, evaluation paradigms are represented to answer the research questions about quality attributes of the studies proposed. In order

to better formulate the research scope and make a comparison between studies, all data items and evaluation paradigms defined in details.

POS tagging techniques and tools

POS tags can be defined as appropriate grammatical descriptors. They give information about the morphological structure of the words (e.g., noun, verb, adjective) in a sentence. Various algorithms are algorithms and tools to predict POS tag of each word are applied in each study. In order to further manipulate, process and subsequently interpret, determining the underlying syntactic structure of the sentence has significant importance (Indurkhya and Damerau, 2010). The parts of a sentence that are considered for further processing, impact of POS tagging is seen on extracting word senses. POS tags and their descriptions are given in Table 1.

Table 2.1. POS tags and their descriptions

POS Tag	Description
ADJ	adjective
ADP	adposition
ADV	adverb
AUX	auxiliary verb
CONJ	coordinating conjunction
DET	determiner
INTJ	interjection
NOUN	noun
NUM	numeral
PART	particle
PRON	pronoun
PROPN	proper noun
PUNCT	punctuation
SCONJ	subordinating conjunction
SYM	symbol
VERB	verb

Dependency analysis techniques and tools

One of the fundamental research topics in natural language processing has been tokenizing natural language sentence into specific types of structures. Diverse relationships between words like object, subject, or other modifiers are determined during dependency analysis phase. Dependency analysis algorithms are based on

dependency structures that is represented with dependency nodes and relations. Dependency graph is generated by representing each token as a node and relations between them. Except the root node, each node is exactly dependent to another (Nivre, 2010). Specifying the partial roles of each token and interpreting them as a whole, play a significant role to achieve meaning of the sentence.

Named entity recognition techniques and tools

Extracted word tokens or phrases are tagged as either person, location, organization or proper noun during named entity recognition. It is considered as the first steps to the semantic level analysis. Mentions are found and resolved in a sentence or text into predefined categories. Named entity recognition is a sub research topic of information extraction. Parsed atomic tokens are located and classified into predefined categories. These categories can also be customized corresponding to the domain requirement. Several recognition algorithms are employed to discover mentions or entities by the applications/studies. Most popular approach while resolving the entities in a sentence or text is utilizing from learning methods and techniques (Celikkaya, Torunoglu, and Eryigit, 2013). Due to the wide usage of named entity recognition for question answering frameworks, this method is selected as a data item for our systematic literature review.

Lexicon corpus

For ontology based approach, a solution is offered by defining a mapping between the words in a sentence or text and vocabulary elements in an ontology. For a specific language, even their verbalization is matched with the elements of the ontology. Lexicon corpus has lexical knowledge which covers the definition of vocabulary elements in an ontology. Additionally, possible different verbalizations and constraints that provide details of syntactic information for a given specific semantic endpoint are all included in a lexicon corpus. Latest studies are emphasizing the importance of building lexicon corpus in an automated manner instead of dealing with manually created lexicon corpus which will result in paying high costs and considerable amount of manual work (Walter, Unger and Cimiano, 2014).

Learning methods

Learning methods mainly deal with improving the results of preprocessing methods and performing inference in the context of question answering frameworks. For the

aim of improving the results of candidate phrase, relations and named entities prediction in a sentence learning methodologies are applied. In order to achieve query intention in a more accurate way, data sets which represent type of phrases, named entities due to predefined categories, relations between words, are employed to train the learning models. Most important point is learning methods give the opportunity to the frameworks not to be language dependent. These methods are based on data dependent and inference the implicit knowledge by using trained data models. Additionally, learning method contributes systems about adapting the language used, processing and expanding the keywords or entities in the question.

Precision, recall and F-measure

Question answering systems as information retrieval systems which are usually evaluated based on the quality metrics precision, recall and f-measure. For a given question q, precision refers to the fraction of relevant answers returned for question q to the total number of answers returned by the system. Recall is the fraction of relevant answers returned for question q to the number of all gold standard answers for q. All retrieved answers are taken account to evaluate precision whereas for recall only relevant are standard answers are considered during evaluation. Last metric is F-measure which is the harmonic mean of precision and recall. General aim of question answering systems directly optimizing the precision and recall values which will also automatically optimize F-measure. Precision, recall and F-measure can be summarized as performance, reliability and quality indicators for the question answering systems. For each question q, precision, recall and F-measure are evaluated as follows (Allam and Haggag, 2012):

$$Precision(q) = \frac{number\ of correct\ system\ answers\ for\ q}{number\ of\ system\ answers\ for\ q} \quad (1)$$

$$Recall(q) = \frac{number\ of correct\ system\ answers\ for\ q}{number\ of\ gold\ standard\ answers\ for\ q} \quad (2)$$

$$F - \mathrm{M}easure(q) = \frac{2 * Precision(q) * Recall(q)}{Precision(q) + Recall(q)} \quad (3)$$

2.2.4. Selection and Evaluation of Studies

Selection of studies

Two main issues are considered while finding answers to the research questions defined in Section 2.1 and select relevant studies for literature review. First one is keyword based search from the databases IEEE, Web of Science, ACM and Science Direct to find candidate studies. In the direction of inclusion/exclusion criteria and data items coverage of the studies, search results are eliminated. Second issue is selecting the comparable studies which narrows the set of studies and direct the study to a way which will exactly match the all research questions of this literature review. Selecting the comparable studies also gives the opportunity to compare the performance of quality and reliability of the studies. Another concern for the comparison is types of methods and tools that are mentioned in data items and evaluation paradigms section (See Section 2.3). Based on these two main issues and research questions inclusion (I) and exclusion (E) criteria are listed below.

I1 – Studies that apply NLP techniques and semantic technologies to question answering frameworks

I2 – Studies that utilize semantic endpoints to generate result for question answering frameworks

I3 – Studies which are comparable with each other in terms of precision, recall and F-measure

E1 – Studies that do not fit the data items described

E2 – Duplicate studies that are published in different sources are excluded

E3 – Studies that are published in the form of abstract, book chapter or tutorial are excluded

Table 2 illustrates the search strings for keyword based database search. Sample search string for IEEE Xplore by using command option in digital library as follows:

("question answering" OR "question answering systems")
AND
("linked data" OR "semantic endpoint" OR "ontology")
AND
("natural language" OR "natural language processing").

Same principle is applied while searching with logical operators for the other databases.

Table 2.2. List for Search Strings

	Search string title	**Author keywords**
String 1	Question answering	question answering OR question answering frameworks OR question answering systems
String 2	Linked data	linked data OR semantic endpoint OR ontology
String 3	Natural language	natural language OR natural language processing OR natural language input

For selection of studies phase, 5 steps are outlined. First step is keyword based database search using the search strings declared in Table 2. Second one is title and abstract filtering which is the first elimination. Titles are reviewed and for some of the studies to decide on elimination abstracts are also reviewed. Then, duplicates are removed from the remained set of studies. After eliminating the duplicates full text screening is performed on the studies. Thereafter, taking inclusion/exclusion criteria into account, one more step is taken to the final set of studies which consists of comparable ones between each other.

Keyword based database search is followed by title and abstract filtering which is the first round selection. 854 studies deemed potentially relevant and retrieved for full text screening and duplicate removal. Afterwards, full text screening and duplicate removing is applied as second round selection and resulted in exclusion of another 582 studies. 272 studies left for full text screening and duplicate removal. Then after inclusion/exclusion criteria filtering is performed and 78 studies are remained. Thereafter, comparable studies are selected in the final round. Finally, 9 studies are considered eligible for the literature review. Steps included for study selection is illustrated on Figure 2.2.

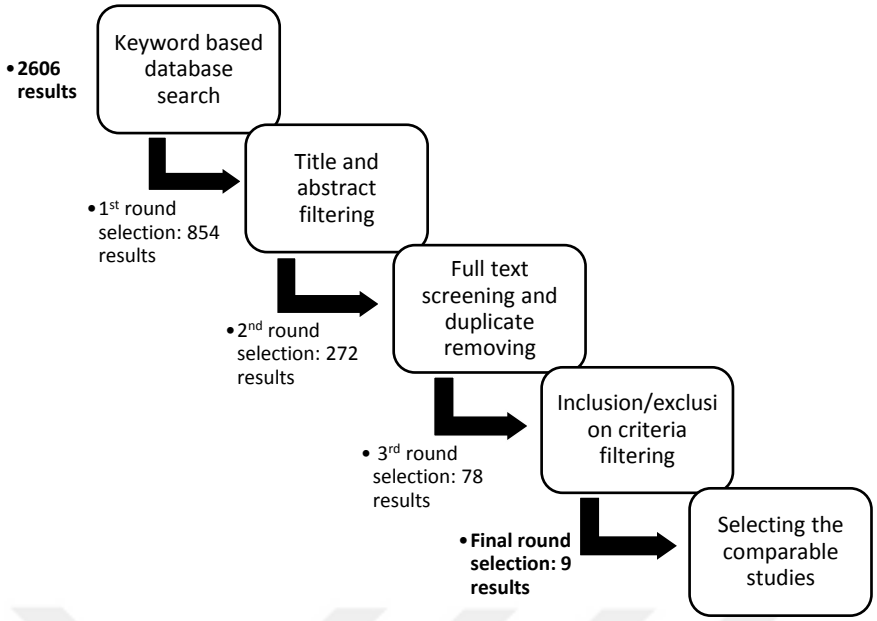


Figure 2.2. Selection of studies

Four different database platforms are used for keyword based search which are listed with details of number of matches and year interval on Table 3. Search is restricted only with the keyword list in Table 2 and by declaring a year interval in terms of publication years.

Table 2.3. Keyword based database search details (initial set of studies)

Database	**Number of Matches**	**Year Interval**
IEEE	396	2010-2017
Web of Science	917	2010-2017
ACM Digital Library	658	2010-2017
ScienceDirect	635	2010-2017

Evaluation of studies

After steps of studies selection is applied 9 studies are remained as final relevant set of studies in literature. Evaluation of these studies and comparison between them is declared in this section. Discussion about the relevant studies in literature is given by considering the research questions and research context to better comprehend methods and tools utilized in these studies.

The first study is represented by Xu et al. (2014) which is a question answering system named Xser. Xser converts natural language question into SPARQL queries to send over DBpedia. Major two problem focus described in their study is recognizing the user intention in an accurate way and mapping the user intention with the

conceptualization represented in the ontology. In order to generate a structured query, semantic objects in DBpedia should match the extracted intention as result of the question analysis. Directed acyclic graph (DAG) in dependency phrase level is utilized to achieve this target. Two main components involved in their study to achieve query intention. In the first phase, phrase detector that is used to recognize candidate semantic phrases which represent entities, categories, relations or variables. Predicate-argument relations among phrases are predicted to be represented as query pattern structures. Then, for the given query pattern structure that represent intention, a structured perceptron model is employed for the second phase. Knowledge base items which represents DBpedia conceptualization are mapped with semantic phrases by solver structured perceptron model jointly. As a summarization, phrase level recognition of query intention is performed and mapping between the phrases and knowledge base concepts is involved in two-phase model of natural language understanding. An example natural language query and converted form to a structured query SPARQL is showed below.

NL Query: *Which country did France colonise?*

SPARQL:
```
SELECT ?x
WHERE
{
?x fb:object.type fb:location.country
fb:en:france fb:colonise ?x
}
```

Xser is tested with QALD-4 [1]dataset for task 1 (multilingual question answering over linked data) which is used as a benchmark to evaluate. Results of the experiment showed that F-measure score of 0.72, an average precision of 0.72 and an average recall of 0.71 over 50 questions is achieved by Xser.

Next study is proposed by Dima (2014) uses both syntactic and semantic information of the natural language question to formulate RDF triples that map with the user intention. Main purpose of Intui3 is to propose an alternative search paradigm to

[1] *https://pub.uni-bielefeld.de/data/2687439*

keyword-based search to overcome the problems about not providing an answer, only providing a list of documents. The name of the search paradigm is Intui3 which is a multilingual question answering platform. Answer is retrieved from a semantic endpoint by converting the question in unstructured format to a structured SPARQL query. Formulated RDF triples form an interpretation which maps the conceptualization to the ontology. Constituent phrases and combination rules for them are evaluated by employing Frege's Principle of Compositionality to construct interpretation of questions. System uses a predicate index and entity index. Predicate index extracted from DBpedia which has 49,714 predicates and entities are all taken from DBpedia. A sample natural language query and outcome generated by Intui3 is shown below.

NL Query: *How many pages does War and Peace have?*

SPARQL:
```
SELECT DISTINCT ?answer
WHERE
{
<http://dbpedia.org/resource/War_and_Peace>
<http://dbpedia.org/property/pages>
?answer .
}
```

Intui3 is tested with QALD-4 dataset for task 1 and results showed that F-measure score of 0.24, an average precision of 0.23 and an average recall of 0.25 over 50 questions in 7 different languages is achieved.

Study of He et al. (2014) is named CASIA@V2 which is a question answering system that takes natural language question and transforms into SPARQL query. Answers are generated over linked data, DBpedia is used as semantic endpoint. Markov Logic Network (Richardson and Domingos, 2006) which serves jointly learning framework is used for phrase detection, grouping of phrases and mapping of phrases with semantic items in DBpedia. First step for their pipeline is question decomposition is carried out to detect the candidate phrases. Afterwards mapping process is launched between the candidate phrases and semantic items in DBpedia. Grouping is used while matching the concepts in Dbpedia with decomposed phrases to generate triples. Finally, triples are used to generate SPARQL query. System architecture of CASIA@V2 is shown in Figure 2.3.

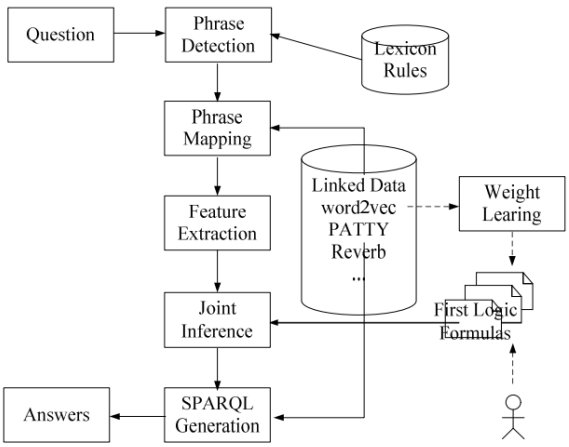


Figure 2.3. System architecture of CASIA@V2 (He et al., 2014)

CASIA@V2 is tested with QALD-4 dataset for task 1 and results showed that F-measure score of 0.36, an average precision of 0.32 and an average recall of 0.40 over 50 questions is achieved.

Next study ISOFT is implemented by Park et al. (2014) which generates SPARQL templates from the natural language questions. Two popular approaches for question answering are combined for ISOFT. First one is knowledge based question answering and the second is information retrieval based question answering. Semantic similarity is the focus intention of this study. By using semantic similarity, questions are mapped with SPARQL templates. Significant words in a query are selected and are combined to perform mapping with defined uniform resource identifiers (URIs) which represent semantic items in the ontology. Significant words addresses that the phrases that has clues which have great contribution to find the answer. They claim that instead of searching URIs for each word for mapping, choosing significant words to focus results in better performance. Overall system architecture is illustrated on Figure 2.4.

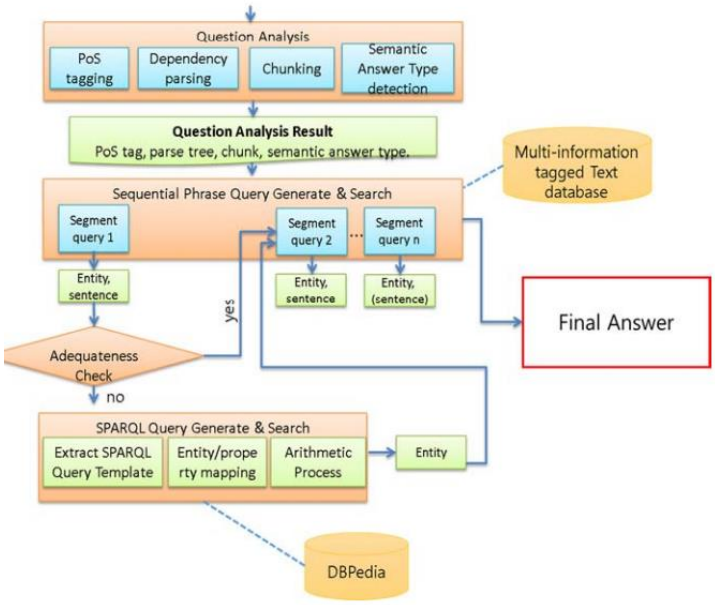


Figure 2.4. Overall system architecture of ISOFT (Park et al., 2014)

ISOFT is tested with QALD-4 dataset for task 1 and results showed that F-measure score of 0.23, an average precision of 0.21 and an average recall of 0.26 over 50 questions is achieved.

Hamon et al. (2014) proposed a study which is named POMELO and composed of NLP methods, semantic resources and RDF triples. Focused domain of this question answering framework is biomedical knowledge. POMELO has four-step method which begins with pre-processing the question with linguistic and semantic annotation. Second step is extracting the abstract of the question which is composed of argument and predicate description. Next step is constructing the SPARQL query using the arguments and predicates. Final step is generating SPARQL query to extract the answer. Architecture of POMELO system is indicated on Figure 2.5.

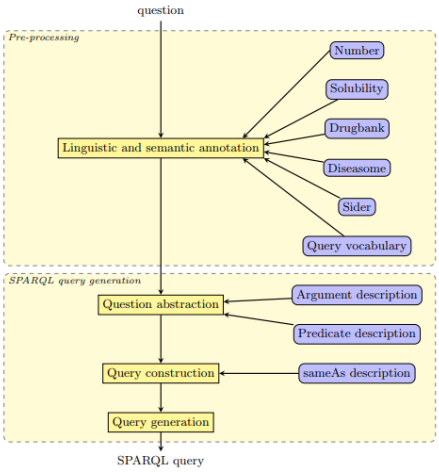


Figure 2.5. Architecture of POMELO (Hamon et al., 2014)

POMELO is tested with QALD-4 dataset for task 2 (Biomedical question answering over interlinked data) and results showed that F-measure score of 0.85, an average precision of 0.82 and an average recall of 0.87 over 25 questions is achieved.

Another study is proposed by Usbeck and Ngomo (2015) which is named HAWK in QALD-5 for hybrid question answering. They implement a pipeline that initiates with the segmentations of words, POS tagging and transition-based dependency parsing. POS tagging is carried out on each tokenized item to use later for semantic annotations. Entities are recognized and linked by using FOX (Speck and Ngomo, 2014) as the next step. FOX is a knowledge extraction framework based on ensemble learning. Semantic role labeling is combined with dependency analysis and noun phrase detection to seize linguistic and semantic relations. Predicate and argument relations are extracted to form a tree during dependency analysis. Architectural overview of HAWK is represented on Figure 2.6.

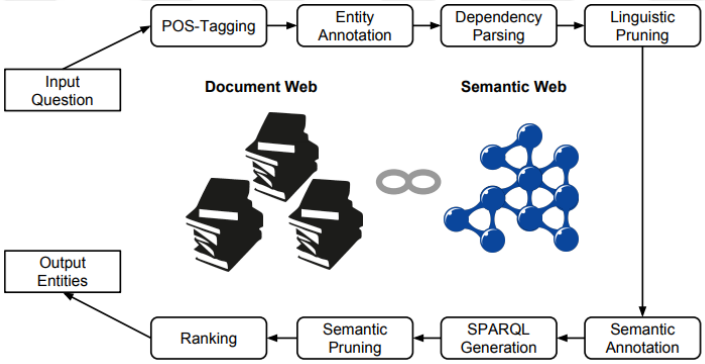
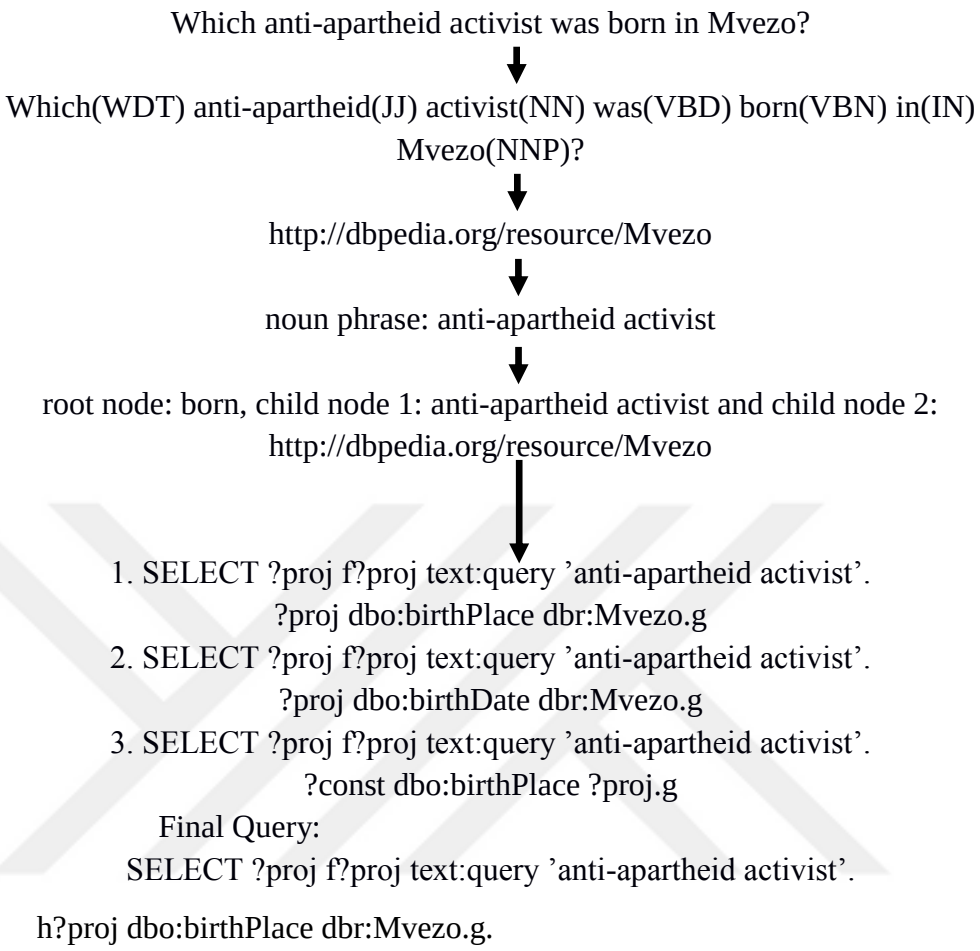


Figure 2.6. Architectural overview of HAWK (Usbeck and Ngomo, 2015)

An algorithm represented with their study identifies noun phrases that are semantically meaningful and not yet annotated by entity recognizer. In order to combine tokens, they prepared a heuristic model derived from POS tag sequences based on benchmark questions (QALD-5). For an effort to improve the F-measure metric of the system two domain expert users implemented the POS tag sequences. Linguistic pruning is performed by considering the types of POS tags. For example: all DEL POS-tags are deleted from a predicate-argument tree which consists of semantically meaningful tokens and entities. Final step for the pipeline is determining the possible verbalizations of ontology items and words to find possible mappings between them. For example: token "born" would match with the properties dbo:birthPlace and dbo:birthdate. From each possible query sample, combinatorial SPARQL graph pattern is generated. Possible queries are eliminated due to a semantic pruning algorithm. Afterwards, cosine similarity between the candidate SPARQL queries and

features of tokens is evaluated to run on a ranking algorithm. Sample output is figured out for each step of pipeline below.



HAWK is tested with QALD-5 [2] dataset and results showed that F-measure score of 0.33, an average precision of 0.33 and an average recall of 0.33 over 10 questions is achieved.

Ruseti et al. (2015) implemented a question answering framework that converts natural language question into SPARQL query. Name of their system is QAnswer, DBpedia is used as knowledge base and queries are executed on semantic endpoint Virtuoso open source platform (Gavitt et al., 2011).

There are two major steps involved in this study. In the first step; DBPedia entities are determined from the text which is followed by the second step that connects them in an appropriate and accurate syntactic dependency type to generate a directed graph.

[2] https://pub.uni-bielefeld.de/data/2900686

Pipeline starts with parsing input question by using Stanford CoreNLP library (Manning et al., 2014). Directed graph with vertices and edges is the output of this parsing process. Vertices represent annotated tokens like lemma and POS tags whereas edges represent collapsed dependencies obtained from Stanford CoreNLP. Named entity recognizer module of Stanford CoreNLP is also utilized to detect numbers and dates. After generating the directed graph, finding DBpedia entities is performed. Properties, types and individuals are the entity types in DBPedia. Entity detection methodology varies corresponding to the entity types. Each type of entity has its own method to trigger the entity detection. Various candidate mappings are obtained which will result in generation of multiple different graphs.

All entities belong other than DBpedia are all ignored for individual detection. Number of Wikipedia pages that have links to the individual's page is an important parameter for importance rank estimation to detect individuals. Determining the individuals can give the opportunity to achieve more specific information while comparing the determination of the properties and types. Even in for some conditions, properties or types can be inferred from the individuals.

For type detection, their study has a Wikipedia based approach. This solution is mainly considering number of times a type exists in a text by label and synonyms of the labels are extracted from WordNet to search for in the text and widen the coverage. For the condition when labels has more than one word, at that point this solution fails and Wikipedia based solution is applied. Wikipedia based approach is mainly considering the strict relation between the Wikipedia and DBpedia entities. Each Wikipedia article has a DBpedia individual tag which has a type formulation. The idea behind to determine type based on the assumption that the first sentence of article includes a short description to determine type.

Property detection is more sophisticated than type and individual detections. Finding where the properties are located in the text and from which types of sentences they are obtained is a complex problem. Syntactic dependencies are useful to solve that problem. Dependencies address the properties in a sentence and generate a pattern for sentence. By using the pattern, they defined a formulation that is composed of both words in path and reachable ones from the path by going on edges and considering individual and type to extract property as an expression. Obtained expression is used to match with all candidate graphs. Finally, a scoring algorithm is applied to find the

graph which has highest score to generate SPARQL. System architecture of QAnswer is showed on Figure 2.7.

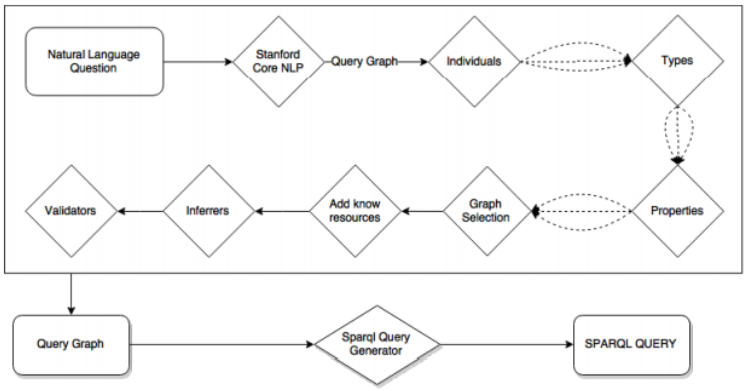


Figure 2.7. System architecture of QAnswer (Ruseti et al., 2015)

QAnswer is tested with QALD-5 dataset and results showed that F-measure score of 0.40, an average precision of 0.46 and an average recall of 0.35 over 50 questions is achieved.

SemGraphQA is a knowledge based question answering system which is developed by Beaumont et al. (2015). SemGraphQA provides simple access to end users without the requirement of knowing how the knowledge is structured or any formal language to access information and transforms natural language input to SPARQL. Graph representation of natural language input is the focus intention of this study. First fundamental aspect, while dealing with the interrelated challenges about representing user intention in a graph representation, is identifying entities, classes of entities, operators and relations. Second aspect is structural and relational form of the entities and how the operators apply.

Target of SemGraphQA is converting a natural language question into semantic graph to present candidate meanings for specified expressions in the question. Unsupervised learning method based on graph conversion is applied for semantic analysis of questions. Nodes represent entities and edges represent relations in the semantic graph. Entities, properties and relations are mapped with pre-defined question phrases. During mapping process, a scoring algorithm which produces relevance score between the phrases and entities/properties/relations to find the best possible match.

Syntactic analysis of questions are performed by Stanford Core NLP (Manning et al., 2014) and a syntactic graph is produced for each natural language input. Nodes are represented with words and edges are represented with syntactic relations in the

produced graph. Considering the conditions if the word is entity or relation in the graph results in the producing the various versions of that syntactic graph to handle ambiguities. Therefore candidate semantic graphs which represent natural language questions are generated. Relevance ranking algorithm is used to select the most relevant graph and SPARQL it represents. System overview with a sample query and generated answer are illustrated on Figure 2.8.

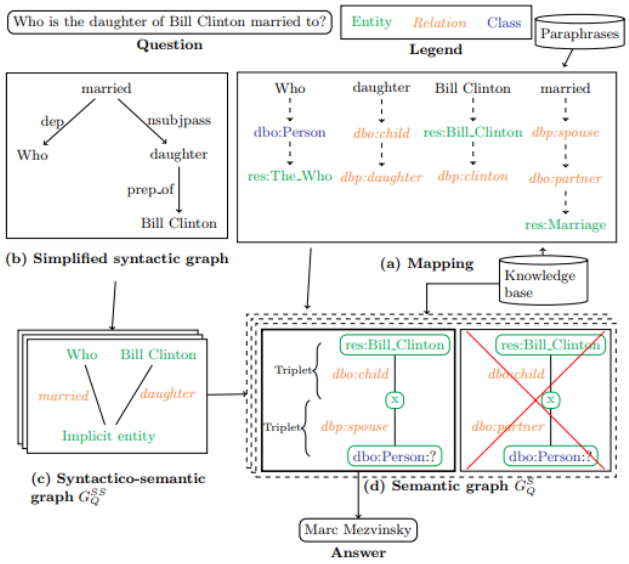


Figure 2.8. System overview of SemGraphQA (Beaumont et al., 2015)

SemGraphQA is tested with QALD-5 dataset and results showed that F-measure score of 0.31, an average precision of 0.31 and an average recall of 0.32 over 50 questions is achieved.

YodaQA (Baudis and Sedivy, 2015) is a modular and open source question answering platform which is designed as a pipeline. Answer production and question analysis strategies utilize from machine learning models to rank the answers by combining diversified knowledge base paradigms. 5 major processing steps of YodaQA can be listed as: question analysis, answer production, answer analysis, answer merging and scoring and successive refining.

Question answering features “Clues”, “Focus” and “LAT (Lexical Answer Type)” described by them to be used for question analysis; POS tagging, dependency parsing and named entity recognition. Clues stands for the keywords in the question to determine content and plays critical role for final query in terms of generating possible candidate answers. Focus is the queried object which locates in the center point to

achieve intention of the input question. Heuristics integrated dependency parsing is used to find focus of the sentence. LAT refers answer type description that is derived from focus.

For answer production, extracted clues are used to set candidate answers. They defined main source of their system is based on the clue search in titles, full text search, document search and concept search. In addition to full text search, they also search in knowledge bases that store RDF triples. Semantic endpoint DBpedia and Freebase is queried for each clue specified as subject and candidate answers are generated for each object in such a triple. Afterwards, generated answers are analyzed by following the principles listed below to decide on LAT.

Answers that use named entity recognizer (OpenNLP which use date, location, money, organization, percentage, person and time pre-defined categories for annotation) produce LAT due to the generated model

- Answers which hold numeric expression have generic quantity LAT
- Extracted focuses are matched with the items in WordNet and DBpedia to generate LATs
- Original property names from knowledge bases are directly accepted as LATs

Duplicates are removed and a classifier is used to classify answers corresponding to their features during answer merging and scoring. Afterwards, an optional step which is successive refining is performed. Generated answers are pruned with taking only top 25 answers to reduce noise in the answer set. Remained set is applied for scoring again. The answer with the highest score is selected as the output of the system. General architecture of YodaQA is illustrated on Figure 2.9.

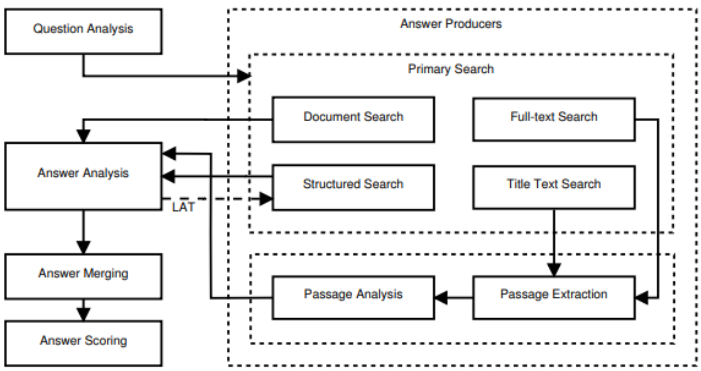


Figure 2.9. General architecture of YodaQA pipeline (Baudis and Sedivy, 2015)

YodaQA is tested with QALD-5 dataset and results showed that F-measure score of 0.26, an average precision of 0.28 and an average recall of 0.25 over 50 questions is achieved.

2.2.5. Comparison of studies

Primary studies are compared below corresponding to the data items and evaluation paradigms specified in section 2.3. In this section, data items based comparison starting from Xser is discussed.

Xser (Xu et al., 2014) is a question answering system that proposes recognizing user intention and generating answers over linked data. They specifically use DBpedia as semantic endpoint. Common challenges for question answering systems as mentioned in the other primary studies are also valid and emphasized for Xser. Proposed pipeline mainly has two main processes. Phrase detection with an algorithm they represented in their study is the initial step of the pipeline. Entity phrases and categories of phrase relations at the syntactic level is handled by this initial step. By using extracted predicate argument dependencies, semantic parsing is performed as the next step.

INTUI3 (Dima, 2014) that is proposed by Dima, combines analysis results NLP frameworks from both SENNA (Collobert et al., 2011) and Stanford CoreNLP (Manning et al., 2014). SENNA which is built in neural network architecture and offer learning algorithms to predict POS tags, NER, semantic role labeling, chunking and syntactic parsing. According to their experiments, they observe that SENNA has a drawback of providing lemma information. They prefer to obtain lemma information from Stanford CoreNLP suite and combine outputs in INTUI3 architecture. Additionally, DBpedia lookup service (Bizer et al., 2009) is used to assign the mapping between DBpedia URIs and given words, word sequences or phrases. Simple string matching techniques are not adequate to find the similar and relevant words or expressions. Therefore, Lucene [3] based index is used for weighted label lookup by combining string similarity techniques and relevance ranking algorithms.

CASIA@V2 (He et al., 2014) is also a question answering framework over DBpedia. Markov Logic Network (Richardson and Domingos, 2006) is used to recognize candidate phrases and generate mapping between these candidates and semantic items

[3] https://lucene.apache.org/core/

in DBpedia. Learning is the significant contributor of their methodology and plays a critical role in their pipeline. In order to predict most accurate and effective patterns to construct semantic triples, weights of the clauses are learned by their system. QALD-3 training data and Free917 data set (Cai and Yates, 2013) for test corpus data for their Stanford CRF based named entity recognition tool. From results it was seen that accuracy rate for QALD-3 data is 51.5% and for Free917 is 23.8%. Accuracy rate stands for the rate of named entities that recognized accurately. In the end, they avoid using a NER tool because of the inadequate accuracy rates towards their target. Instead, all n-grams are extracted as candidate phrases and use a set of rules (like checking length of tokens, capitalization tokens cannot be split etc.) to categorize by applying their proposed algorithm. word2vec (Mikolov et al., 2013) tool is employed for computation of similarity between the phrase and the class in DBpedia. For each phrase, top-n best match, most similar classes are returned as result. Beside that tools all mentioned here, ReVerb (Fader, Soderland and Etzioni, 2011) and PATTY (Nakashole, Weikum and Suchanek, 2012) provide resources for textual patterns to represent binary relations between entities for bigrams. Question classification is performed to detect the question type (e.g., “who”, “what”, “when”, “where”, “how”) in ISOFT (Park et al., 2014). After question is classified, input question is morphologically analyzed by using ClearNLP (Choi and Palmer, 2011) as semantic role labeler and OpenNLP (Morton et al., 2005) for chunking. Additionally, resource disambiguation and named entity recognition is handled by AIDA (Yosef et al., 2011). For the condition that there exist more than one named entities or noun-phrases, dependency parsing is applied as a rule defined in ISOFT. Head of each word and relations between them are detected to achieve whole meaning by generating a predicate. Results obtained during natural language analysis are used for further analysis of input question and several rules are derived from results to extract slots to generate matched SPARQL templates.

POMELO that is proposed by Hamon et al. (2014) is a question answering system in biomedical domain. For POMELO input questions are annotated by using NLP methods as preprocessing step of their pipeline. TreeTagger (Schmid, 2013) tool, which is based on decision tree methods, is utilized for numerical entity tagging, word segmentation, POS tagging and lemmatization. TermTagger [4] Perl module is used to

[4] http://search.cpan.org/~thhamon/Alvis-TfermTagger/

extend the types of predefined categories for named entity recognition (like recognizing disease names, side effects etc.) to increase the coverage of semantic resources. Beside TreeTagger and TermTagger, YATEA which is implemented in the scope of ALVIS project (Aubin et al., 2006), is used to tag terminological entities to improve the coverage for terminological content.

Hybrid question answering system HAWK combines linked data and textual data to serve answers. In the pipeline proposed, initial input processing step is tokenizing and POS tagging. Results from transition based dependency parsing and POS tagging is obtained from ClearNLP (Choi and Palmer, 2011). Named entity recognition and linking is performed by FOX (Speck and Ngomo, 2014) which is an ensemble learning based framework for knowledge extraction. Word groups or expressions that are not recognized by the entity annotator and have semantically critical contribution to the meaning of the sentence are determined by their own algorithm in HAWK.

QAnswer system uses Stanford CoreNLP library (Manning et al., 2014) for syntactic parsing and POS tagging. Dependencies between words are also obtained by using dependency parsing module of this library. Numbers and dates are annotated with named entity recognition with CoreNLP library. They used their own algorithm to match annotated entities with DBpedia entities. Semantic items are mapped with entity annotation results. WordNet is used to extend the coverage with synonym definitions while annotating entities.

SemGraphQA which is a knowledge based question answering system proposed in the study of Beaumont et al. (2015). Unsupervised method is introduced for semantic analysis of questions. Fundamental focus of the study is performing a transformation of natural language input to a graph representation to be further converted to SPARQL as the next step. For preprocessing of input question, syntactic analysis is applied with Stanford CoreNLP library (Manning et al., 2014). Additionally, entity annotation is performed by DBpedia Spotlight.

In the study of Baudis and Sedivy (2015) an open source pipeline question answering system which is named YodaQA is introduced. The pipeline is built on suite of NLP analysis tools and methods to process the question and generate answer. Stanford CoreNLP (Manning et al., 2014) is used for segmentation, POS tagging and name entity recognition. In addition to this library, they utilize from OpenNLP (Morton et al., 2005) to extend coverage by combining outputs.

According to the discussion given for each study a comparison is drawn based on data items and evaluation paradigms.

INTUI3, ISOFT, POMELO and CASIA@V2 competed in the QALD-4 challenge, whereas HAWK, QAnswer, SemGraphQA and YodaQA belong to the QALD-5 challenge. XSER joined both for QALD-4 and QALD-5 challenges. Final result show that most successful system which means achieving the meaning of input question, understanding user intention and generating accurate answers is Xser. QALD-4 and QALD-5 test datasets are used to achieve evaluation paradigms precision, recall and F-measure. Test questions of QALD-4 are slightly different from QALD-5 questions which can be assumed that they are compared by using similar content. This difference is ignored while performing comparison.

2.3. Conclusion

Major concern for computational question answering is producing accurate and reliable answers for the input questions. Improving the values for precision, recall and F-measure as performance and quality indicators is a continuous effort for question answering systems. Following this reason, question answering systems, use, integrate and combine several techniques and tools. Studies that combine both natural language processing techniques and linked data technologies summarized systematically in the literature review section. Steps described in Figure 1 are the main parts discussed for this section. Identifying research questions as initial step directed the way for conducting a systematic review, enlightening the progress of the study and defining the focus intention of the literature review section. In addition to this, research questions help to identify data items and define the context and boundaries of the literature review. For more precisely determining the boundaries of review, inclusion/exclusion criteria filtering is also applied. Data items are evaluation paradigms are defined to better formulate the comparison between the studies in literature. From an objective point of view, related studies in literature are discussed.

Table 2.4. Comparison of studies

System	POS Tagging	Dependency Analysis	Name Entity Recognition	Lexicon Corpus	Learning Mechanism	Precision, Recall, F-Measure
INTUI3	✓	✓	✓			0.23, 0.25, 0.24
ISOFT	✓	✓	✓			0.21, 0.26, 0.23
POMELO*	✓		✓			0.82, 0.87, 0.85
XSER	✓	✓	✓		✓	(QALD4) -> 0.72, 0.71, 0.72 (QALD5) ->0.74, 0.72, 0.73
CASIA@V2	✓	✓		✓	✓	0.32, 0.40, 0.36
HAWK*	✓	✓	✓			0.33, 0.33, 0.33
QANSWER	✓	✓	✓	✓		0.46, 0.35, 0.40
SEMGRAPHQA	✓	✓	✓	✓		0.31, 0.32, 0.31
YODAQA	✓	✓	✓	✓	✓	0.28, 0.25, 0.26

CHAPTER 3
GEOGRAPHIC TURKISH QUESTION ANSWERING FRAMEWORK (GEO-TR)

This thesis study proposes a question answering framework over linked data for the given Turkish input sentence. Fundamental modules and their submodules are illustrated in Figure 3.1. This framework is designed and implemented to process two types of questions. Type 1 is informative questions. “Türkiye’nin komşuları hangi ülkelerdir? (Which countries are the neigbours of Turkey?)”, “Ege Bölgesi'ndeki şehirlerin nüfuslarını gösterir misin ? (Can you show the populations of cities in Aegean Region?)”, “Akdeniz Bölgesi’nde hangi göller bulunmaktadır? (Which lakes are in the Mediterranean Region?)” can be given as examples for Type 1 questions. Informative questions can hold quantitative analysis required expressions like “Antalya’nın nüfusu kaçtır? (What is the population of Antalya?)”, “Harran ovasının yüzölçümü kaçtır? (What is the land area of Harran plain?)”. Type 2 represents the sentences which requires quantitative reasoning. “Türkiye’nin en kalabalık ili hangisidir? (Which is the most crowded city in Turkey?)”, “İzmir’de kaç tane dağ vardır ? (How many mountains are there in Izmir?)”, “Türkiye’nin en az tuzlu denizinin adı nedir? (What is the name of the sea of Turkey that has least salinity?)” can be given as examples for Type 2 questions. Question pre-processing is triggered with analyzing and disambiguation of the sentence morphologically. Named entities and dependencies between words are obtained with named entity recognition and dependency analysis. After analyzing the given input sentence, query formulation is one step forward to generate the answer. Answer type is decided by using the outputs generated during question pre-processing. Classification model is employed for the Type 2 sentences that requires quantitative reasoning to be answered. Then, relations are extracted and final query is formulated for both of these conditions. Generated final query is executed on the developed ontology GEO-TR and answer is returned as the output result of the question answering framework.

Section 3 is structured as follows: Firstly, techniques of question pre-processing; morphological analysis and disambiguation, named entity recognition and dependency parsing are described to better understand the NLP pipeline. Then, by using sample input sentences, outputs are generated and details about NLP pipeline are given. Afterwards, query formulation and SPARQL formulations are described. An ontology is developed within the scope of this thesis. Details about the ontology development is presented. Comparison of two approaches while answering the question over linked data is discussed in experimental study and comparison section. Finally, conclusion and future works are suggested.

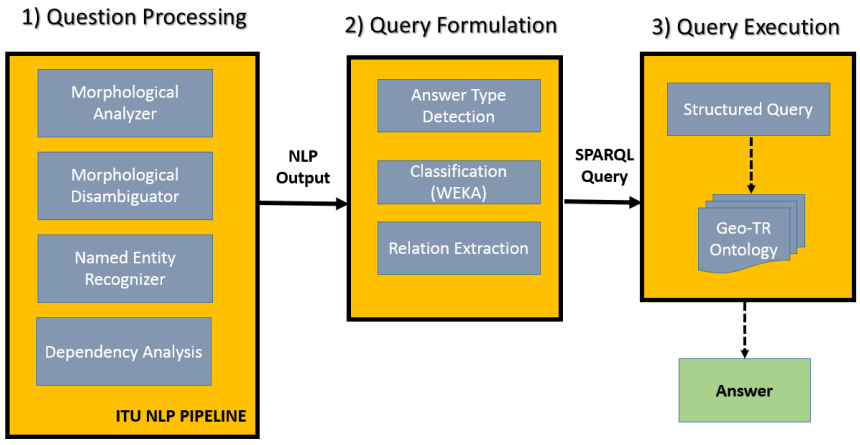


Figure 3.1. Question answering framework over GEO-TR ontology

3.1 Question Pre-processing

For a human brain to understand the question, question is processed to understand terms involved in the sentence, the relations between the words to achieve meaning and deciding on the answer type to generate the final answer. Same principle is modelled for the question answering systems and frameworks. Considering the steps in the human brain while processing an expression, question processing is mainly based on the combination of syntactic and semantic analysis of the expressions. Extracting the focus of the sentence, named entities (if exist), possible relations between the focus and given entities, eliminating the stop words which do not contribute to the meaning and deciding the answer type draw up the processes of question answering frameworks. This thesis presents question answering framework over linked data in pipeline form for question sentences in Turkish for geographical

domain.

As known that by mostly Turkish NLP researchers and linguistic experts, Turkish is really different from other languages morphologically. Turkish is a free constituent order language, and the word order can be grouped as ordered sentence or unordered sentence. As an additional morphological difference from other languages, Turkish is an agglutinative language. In Turkish, there are two groups of suffixes that are defined as constructive suffixes and inflectional suffixes. Constructive suffixes form a new dictionary-word from an old one. An example can be given as "-mek, mak, -van" suffixes and the usage is "yap-mak, yay-van". Inflectional suffixes allow a dictionary-word to take its proper place in a sentence. An example can be given as "-ca, -ce, -ça, -çe" suffixes and the usage is "kısaca, kolayca, sadece" (Erguvanli and Taylan, 1984). By using these suffixes, it is possible to make various combinations which has different meanings for a given root of a word. These features of the Turkish make this language morphologically rich and also really hard to apply traditional English based methods for NLP.

For this study, atomic NLP components of Turkish NLP Pipeline (Eryigit, 2014) is utilized by using web API. Turkish NLP Pipeline is a NLP platform developed by natural language processing group of Istanbul Technical University (ITU) which operates as a SaaS (Software as a Service) and available at tools.nlp.itu.edu.tr. They also provide a Web API to implement higher level applications by using the components of the platform. Output generated by the atomic components which are named: "Morphological Analyzer", "Morphological Disambiguator", "Named Entity Recognizer" and "Dependency Parser" are used for this study. How these components are used and sequential steps in the question processing to determine the answer type are explained in this section.

3.1.1 Morphological Analysis and Disambiguation

Morphologically analyzing a question is essential to determine the morphologically tag of each token (word) to further manipulate and enrich the sentence. Morphological analyzer of the ITU Turkish NLP Web Service is based on two-level analyzing model based on a lexicon of word lemmata with over 49321 entries. Main analyzer uses flag diacritics for Turkish to handle the morphological differences in Turkish. Turkish is an agglutinative language with many exceptions in terms of phonetic and morphological rules. Therefore, they use flag diacritics to handle these exceptions.

Another analyzer for the unknown words demonstrates the use of affix stripping to find word lemmata by recursively removing affixes without having an additional lexicon. Further details of their methodology expressed in (Sahin, Sulubacak and Eryigit, 2013). Output of the morphological analyzer for a sample question is indicated below:

Ankara iline komşu olan illeri gösterir misin ?

Morphological analyzer output:

Ankara+Noun+Prop+A3sg+Pnon+Nom
il+Noun+A3sg+P2sg+Dat il+Noun+A3sg+P3sg+Dat
komşu+Adj komşu+Noun+NAdj+A3sg+Pnon+Nom
ol+Verb+Pos^DB+Adj+PresPart
ol+Verb+Pos^DB+Adj+PresPart^DB+Noun+Zero+A3sg+Pnon+Nom
il+Noun+A3pl+P3pl+Nom il+Noun+A3pl+Pnon+Acc
il+Noun+A3pl+P3sg+Nom il+Noun+A3sg+P3pl+Nom
göster+Verb+Pos+Aor+A3sg göster+Verb+Pos^DB+Adj+AorPart
mi+Postp+Ques+Pres+A2sg
?+Punc

As seen from the output, some tokens have more than one possible analysis results. For this kind of tokens, disambiguation is required. Morphological disambiguator component of the ITU web service gets the output of the morphological analyzer as input and generates the disambiguated form of the parsed expression. Here for this example, for the tokens “il” and “ol” is referred to two possible POS tag sequence. Disambiguation applied form of the sentence is indicated below:

Ankara iline komşu olan illeri gösterir misin ?

Morphological disambiguator output:

Ankara Ankara+Noun+Prop+A3sg+Pnon+Nom
iline il+Noun+A3sg+P3sg+Dat
komşu komşu+Adj
olan ol+Verb+Pos^DB+Adj+PresPart
illeri il+Noun+A3pl+Pnon+Acc
gösterir göster+Verb+Pos+Aor+A3sg
misin mi+Postp+Ques+Pres+A2sg
? ?+Punc

Morphologically disambiguated output is used to identify the underlying structure of the sequence of words by predicting POS tags for each word. Extracting the word senses for only the parts which are taken into consideration is seen as the impact of POS tagging for further processing to semantically enrich the sentence.

3.1.2 Named Entity Recognition

Further processing effort continues with the named entity recognition in the pipeline.

Named entity recognition discovers entities by extracting and tagging the noun phrases as either person, organization, location, date, time or money. Resolving the mentions in a sentence or free text is critical for semantic level analysis and decision of the answer type. Atomic tokens (parsed words) are located and classified into predefined categories. Predefined categories can be extended according to the context of the information requirement. Several NER techniques serve for the applications. But, most of them are well-studied for English unlike the ITU NLP tool that focuses on recognizing entities for Turkish which is a morphologically rich and less-studied language. Methodology used for the ITU Turkish NLP Web Service is based on Conditional Random Fields (CRF) technique for statistical modelling (Lafferty, McCallum, and Pereira, 2001). Rich morphological structure of the Turkish language represented as features to CRFs with the use of some basic and generative gazetteers. Instead of choosing Hidden Markov Model (HMM), stochastic grammars and maximum entropy Markov models (MEMMs) which are popular techniques for the named entity recognition task, they claim that CRFs offer several advantages in terms of including rich and overlapping features focusing on conditional distribution. Details of their methodology described in their study (Seker and Eryigit, 2012). Another study is utilized for ITU web service to extend CRF based name entity recognition model for Turkish well-formed texts and sentences. They perform re-annotation on the available data sets to increase the coverage of the named entity types and a brand new dataset from Web 2.0. Currently, seven types of entity (person, location, organization, date, time, money, percentage) are supported in the ITU web service. The system introduced in the study uses extensive morphological information which is reported that has significant positive influence and improvement on NER process and achieved the exact match F1 (F-measure; rate of accuracy evaluated using precision and recall values) score of 92% on a dataset collected from Turkish news articles and 65% on different datasets collected from Web 2.0. Details of their methodology described in their study (Seker and Eryigit, 2017). Morphological disambiguator output is formatted to be used in name entity recognizer as input by adding the pre-defined tags by the web service. Output of NER module for disambiguated form of the sentence is showed below.

Ankara iline komşu olan illeri gösterir misin ?

Named entity recognizer output:

Ankara Ankara+Noun+Prop+A3sg+Pnon+Nom	*B-LOCATION*
iline il+Noun+A3sg+P3sg+Dat	*O*
komşu komşu+Adj	*O*
olan ol+Verb+Pos^DB+Adj+PresPart	*O*
illeri il+Noun+A3pl+Pnon+Acc	*O*
gösterir göster+Verb+Pos+Aor+A3sg	*O*
misin mi+Postp+Ques+Pres+A2sg	*O*
? ?+Punc	*O*

"Ankara" is tagged as B-LOCATION which recognized as a first location identifier. Prefixes in the output I-O-B represents in, out and begin. First token of a named entity is annotated with a "B-" prefix while other tokens in the same named entity are annotated with an "I-" prefix. Tokens that are not part of any named entity are tagged with the label "O". For example:

Akdeniz Bölgesi'ndeki şehirleri listeler misin ?

Akdeniz Akdeniz+Noun+Prop+A3sg+Pnon+Nom	*B-LOCATION*
Bölgesi'ndeki bölge+Noun+A3sg+P3sg+Loc^DB+Adj+Rel	*I-LOCATION*
şehirleri şehir+Noun+A3pl+Pnon+Acc	*O*
listeler liste+Noun+A3pl+Pnon+Nom	*O*
misin mi+Postp+Ques+Pres+A2sg	*O*
? ?+Punc	*O*

3.1.3. Dependency Analysis

Pipeline continues with extracting the roles of every token in the sentence in terms of determining the various relations between them like tagging as object, subject, verb or other modifiers. Relational tagging between words with grammatical structures is called dependency analysis. Tokenizing the sentences into certain structures has been one of the fundamental topics in NLP. Dependency analysis algorithms are generally designed upon a dependency graph that is composed of dependency nodes and relations. In a dependency graph, except root node, every node is represented with every single word token which is dependent in exactly one different node (Nivre, 2010). The information about who is doing what to whom which is critical to achieve the semantics is provided by dependency analysis. Partial role of words in a sentence is considered as one step forward to extract meaning (Shehata, Karray and Kamel, 2007). Tags such as SUBJECT, OBJECT, SENTENCE, MODIFIER, CLASSIFIER, POSSESSOR, and etc. defined in Turkish Dependency TreeBank are used by the

dependency parser in Turkish NLP Pipeline (Eryigit, 2012; Eryigit, Nivre and Oflazer, 2008).

Before demonstrating the results of dependency analysis for the sample sentence, CoNLL-X which is a shared task on multilingual dependency parsing should be introduced first. Conference on Computational Natural Language Learning (CoNLL) (Buchholz and Marsi, 2006) has a shared task for the participants to train and test their systems on exactly the same datasets to make the systems comparable with each other. The tenth CoNLL featured a shared task on multilingual dependency parsing. Common input format which is named CoNLL-X is used to train multilingual dependency parsers like MaltParser (Nivre et al., 2007). CoNLL-X standard input has simple column-based format. Each word in a sentence is represented by one line that consists of 10 fields which addresses the information of morphological analysis result of the corresponding token. A sample input sentence ("Türkiye'deki coğrafi bölgeleri sıralar mısın?" -Can you list the geographical regions in Turkey?-) in the CoNLL-X format is showed in Table 3.1 below.

Table 3.1. CoNLL-X format for a sample input sentence

ID	FORM	LEMMA	CPOSTAG	POSTAG	FEATS	HEAD	DEPREL	PHEAD	PDEPREL
1	Türkiye'deki	Türkiye	Noun	Noun	Prop|A3sg|Pnon|Loc^DB|Adj|Rel	_	4	_	MODIFIER
2	coğrafi	coğrafi	Adj	Adj	_	_	3	_	MODIFIER
3	bölgeleri	bölge	Noun	Noun	A3pl|P3pl|Nom	_	4	_	SUBJECT
4	sıralar	sırala	Verb	Verb	Pos|Aor|A3sg	_	5	_	ARGUMENT
5	mısın	mı	Postp	Postp	Ques|Pres|A2sg	_	0	_	PREDICATE
6	?	?	Punc	Punc	_	_	5	_	PUNCTUATION

ID represents the token counter that starts from 1 for each new sentence. **FORM** is the word form in originally as used in the sentence or punctuation symbol. **LEMMA** is the stem of a word form or an underscore if not available. Lemma depends on a particular treebank. **CPOSTAG** stands for the coarse-grained part-of-speech tag from the tagset that is defined in a particular treebank. **POSTAG** shows the fine-grained part-of-speech tag from defined tagset like CPOSTAG. If no POSTAG is available from the treebank used, these two columns are identical and mostly they address the same POS tag. Unordered set of syntactic and/or morphological structures are represented in the column **FEATS**. If any syntactic/morphological feature is not found,

an underscore is the indicator. Vertical bars (|) are used to separate the set members. **HEAD** and **PHEAD** values are not tagged for the input and output CoNLL format for MaltParser (Nivre et al., 2007), so these two columns are ignored for this study. **DEPREL** is the token number which the current token has dependency relation with. The dependency relation of a token with 0 is the root node. **PDEPREL** defines name of the dependency type with the token number specified in the DEPREL column.

CoNLL-X formatted input of the sample sentence ("Ankara iline komşu olan illeri gösterir misin?"- Can you show the cities that has neighbourhood to Ankara?-) to be prepared for dependency analysis is indicated on Table 3.2:

Table 3.2. CoNLL-X formatted input of the sample sentence

ID	FORM	LEMMA	CPOSTAG	POSTAG	FEATS	HEAD	DEPREL	PHEAD	PDEPREL
1	Ankara	Ankara	Noun	Noun	Prop|A3sg|Pnon|Nom	_	_	_	_
2	iline	il	Noun	Noun	A3sg|P3sg|Dat	_	_	_	_
3	komşu	komşu	Adj	Adj	_	_	_	_	_
4	olan	ol	Verb	Verb	Pos^DB|Adj|PresPart	_	_	_	_
5	illeri	il	Noun	Noun	A3pl|Pnon|Acc	_	_	_	_
6	gösterir	göster	Verb	Verb	Pos|Aor|A3sg	_	_	_	_
7	misin	mi	Postp	Postp	Ques|Pres|A2sg	_	_	_	_
8	?	?	Punc	Punc	_	_	_	_	_

Pre-processed CoNLL-X input is sent to dependency analysis module to find the relations between each token. Dependency analysis output of the sample case sentence ("Ankara iline komşu olan illeri gösterir misin?"- Can you show the cities that has neighbourhood to Ankara?-) is showed in Table 3.3.

Table 3.3. Dependency analysis output of the sample sentence

ID	FORM	LEMMA	CPOSTAG	POSTAG	FEATS	HEAD	DEPREL	PHEAD	PDEPREL
1	Ankara	Ankara	Noun	Noun	Prop|A3sg|Pnon|Nom	_	2	_	POSSESSOR
2	iline	il	Noun	Noun	A3sg|P3sg|Dat	_	4	_	MODIFIER
3	komşu	komşu	Adj	Adj	_	_	4	_	MODIFIER
4	olan	ol	Verb	Verb	Pos^DB|Adj|PresPart	_	6	_	MODIFIER
5	illeri	il	Noun	Noun	A3pl|Pnon|Acc	_	6	_	OBJECT
6	gösterir	göster	Verb	Verb	Pos|Aor|A3sg	_	7	_	ARGUMENT
7	misin	mi	Postp	Postp	Ques|Pres|A2sg	_	0	_	PREDICATE
8	?	?	Punc	Punc	_	_	7	_	PUNCTUATION

3.2. NLP Pipeline

Pre-processing techniques described in Section 3.1 are utilized to construct a pipeline

by sequentially performing of pre-processing methods. Algorithm 1 designed and implemented that represents the pipeline to translate a natural language question to a SPARQL query. Spotters are implemented that converts pre-processed input into valid formats that are appropriate for ITU Turkish NLP Web Service. Only method that accepts raw data input as natural language sentence is morphological analyzer (createMorphAnalyzerOutput). For each other method, specific formats are converted and prepared for the next sequential processing step by the implemented spotters. In order to better understand spotters, processing of a sample question by spotter methods which are **createDisambiguatorInput**, **convertToConll** and **createNerInput**, is demonstrated below. For a given question: "Manisa şehrinin çevresinde hangi şehirler konumlanır? (Which cities are located nearby Manisa?)", outputs are indicated according to the sequence between steps 1 and 7 in Algorithm 1.

Morphological analyzer output: (Algorithm 1 - Step 1)

Manisa+Noun+Prop+A3sg+Pnon+Nom
şehir+Noun+A3sg+P2sg+Gen ***şehir***+Noun+A3sg+P3sg+Gen
çevre+Noun+A3sg+P3sg+Loc
hangi+Pron+Ques+A3sg+Pnon+Nom ***hangi***+Adj
hangi+Noun+NAdj+A3sg+Pnon+Nom
şehir+Noun+A3pl+Pnon+Nom
şehir+Noun+A3sg+Pnon+Nom^DB+Verb+Zero+Pres+A3pl
konumla+Verb+Pass+Pos+Aor+A3sg
konumla+Verb+Pass+Pos^DB+Adj+AorPart
konum+Noun+A3sg+Pnon+Nom^DB+Verb+Acquire+Pos+Aor+A3sg
konum+Noun+A3sg+Pnon+Nom^DB+Verb+Acquire+Pos^DB+Adj+AorPart
?+Punc

At this step, a rule based two-level Turkish morphological analyzer by Sahin, Sulubacak and Eryigit (2013) which is based on a lexicon word lemmata with over 49321 entries is employed. Main analyzer (first level) figures out the use of flag diacritics for Turkish. Turkish is an agglutinative language which results in many exceptions to phonetic and morphological rules so using flag diacritics is useful to handle these exceptions. For the unknown word analyzer (second level), any extra lexicon is not utilized. Recursively removing the affixes of the word by affix stripping is used to find word lemmata.

Disambiguator input: (Algorithm 1 - Step 2)

<S> <S>+BSTag

Manisa Manisa+Noun+Prop+A3sg+Pnon+Nom

şehrinin şehir+Noun+A3sg+P2sg+Gen şehir+Noun+A3sg+P3sg+Gen

çevresinde çevre+Noun+A3sg+P3sg+Loc

hangi hangi+Pron+Ques+A3sg+Pnon+Nom hangi+Adj

hangi+Noun+NAdj+A3sg+Pnon+Nom

şehirler şehir+Noun+A3pl+Pnon+Nom

şehir+Noun+A3sg+Pnon+Nom^DB+Verb+Zero+Pres+A3pl

konumlanır konumla+Verb+Pass+Pos+Aor+A3sg

konumla+Verb+Pass+Pos^DB+Adj+AorPart

konum+Noun+A3sg+Pnon+Nom^DB+Verb+Acquire+Pos+Aor+A3sg

konum+Noun+A3sg+Pnon+Nom^DB+Verb+Acquire+Pos^DB+Adj+AorPart

? ?+Punc

</S> </S>+ ESTag

Annotated output of morphological analyzer is converted to be prepared for disambiguation step. Spotter method only adds the tag "***<S> <S>+BSTag***" at the beginning and end of the input.

Disambiguator output: (Algorithm 1 - Step 3)

<S> <S>+BSTag

Manisa Manisa+Noun+Prop+A3sg+Pnon+Nom

şehrinin şehir+Noun+A3sg+P3sg+Gen

çevresinde çevre+Noun+A3sg+P3sg+Loc

hangi hangi+Adj

şehirler şehir+Noun+A3pl+Pnon+Nom

konumlanır konumla+Verb+Pass+Pos+Aor+A3sg

? ?+Punc

</S> </S>+ ESTag

Candidates of the morphological analysis output for each token is disambiguated to be prepared for detecting the dependencies between words and named entities in the sentence. Disambiguator output is converted to the CoNLL format which is the valid input form of MaltParser (Nivre et al., 2007) mentioned in Section 3.1.3. Spotter method **convertToConll** firstly removes the added tags (<S> <S>+BSTag) in Algorithm 1 – Step 2. Then, each line of output is splitted from the blanks. First part (ex: Manisa) is taken as the first column and in the second part (Manisa+Noun+Prop+A3sg+Pnon+Nom) "+" string is replaced by string "|" (Manisa|Noun|Prop|A3sg|Pnon|Nom). First two "|" symbols are replaced with string "\t", second column (Noun) is duplicated by adding the string "\t" between them and the others remained same (Manisa ManisaNoun Noun Prop|A3sg|Pnon|Nom).

Finally string "_" are added to each line to fulfill the number of columns in CoNLL format (this column is ignored by MaltParser and remained blank) and number of each token is added at the beginning of each line.

CoNLL output: (Algorithm 1 - Step 4)

1	Manisa	Manisa	Noun	Noun	Prop|A3sg|Pnon|Nom	_
2	şehrinin	şehir	Noun	Noun	A3sg|P3sg|Gen	_
3	çevresinde	çevre	Noun	Noun	A3sg|P3sg|Loc	_
4	hangi	hangi	Adj	Adj	_	_
5	şehirler	şehir	Noun	Noun	A3pl|Pnon|Nom	_
6	konumlanır	konumla	Verb	Verb	Pass|Pos|Aor|A3sg	_
7	?	?	Punc	Punc	_	_

NER input: (Algorithm 1 - Step 5)

<DOC> <DOC>+BDTag<S> <S>+BSTag

Manisa Manisa+Noun+Prop+A3sg+Pnon+Nom

şehrinin şehir+Noun+A3sg+P3sg+Gen

çevresinde çevre+Noun+A3sg+P3sg+Loc

hangi hangi+Adj

şehirler şehir+Noun+A3pl+Pnon+Nom

konumlanır konumla+Verb+Pass+Pos+Aor+A3sg

? ?+Punc

</S> </S>+ ESTag<DOC> <DOC>+EDTag

Annotated disambiguator output is converted to NER input by adding the tags ***<DOC> <DOC>+BDTag*** at the beginning and ***<DOC> <DOC>+EDTag*** at the end of the input. Converted new form of output is sent to NER layer and named entities are annotated which is showed below for the sample sentence. Seker and Eryigit (2012) designed a named entity recognizer which is extended by Seker and Eryigit (2017) to increase the coverage of the named tags. Current API serves the latest version of tags.

NER output: (Algorithm 1 - Step 6)

<DOC> <DOC>+BDTag<S> <S>+BSTag O

Manisa Manisa+Noun+Prop+A3sg+Pnon+Nom ***B-LOCATION***

şehrinin şehir+Noun+A3sg+P3sg+Gen O

çevresinde çevre+Noun+A3sg+P3sg+Loc O

hangi hangi+Adj O

şehirler şehir+Noun+A3pl+Pnon+Nom O

konumlanır konumla+Verb+Pass+Pos+Aor+A3sg O

? ?+Punc O

</S> </S>+ ESTag<DOC> <DOC>+EDTag

Converted dependency analysis input is sent to dependency analysis module and the result for the sample sentence is demonstrated below.

Dependency analysis output: (Algorithm 1 - Step 7)

1 Manisa Manisa Noun Noun Prop|A3sg|Pnon|Nom _ 2 POSSESSOR

2 şehrinin şehir Noun Noun A3sg|P3sg|Gen _ 3 POSSESSOR

3 çevresinde çevre Noun Noun A3sg|P3sg|Loc _ 6 MODIFIER

4 hangi hangi Adj Adj _ _ 5 MODIFIER

5 şehirler şehir Noun Noun A3pl|Pnon|Nom _ 6 SUBJECT

6 konumlanır konumla Verb Verb Pass|Pos|Aor|A3sg _ 0 PREDICATE

7 ? ? Punc Punc _ _ 6 PUNCTUATION

In Step 8, **checkQuantitativeAnalysis** method decides whether quantitative reasoning is required or not, in other words deciding for the given analyzed sentence is Type 1 or Type 2 by using NER and dependency analysis output. For the given sample sentence, isQuantitative parameter will be assigned to false which means given question is Type 1 and so algorithm moves to Step 14 to generate final SPARQL query. Details for the methods **checkQuantitativeAnalysis, generateSparqlComponents, generateSparqlQuantitative** and **generateSparql** are described in Section 3.3.2.

Algorithm 1

Algorithm designed and implemented to translate a natural language sentence in Turkish to SPARQL query by processing in a pipeline with using NLP techniques (morphologically analyzing, disambiguating and named entity recognition is performed)

Require: sentence s

agenda: generate query by using NLP output

Ensure: final_query

1: morphAnalyzerOutput = pipeline.createMorphAnalyzerOutput(s)

2: disambiguatorInput = pipeline.createDisambiguatorInput(morphAnalyzerOutput,s)

3: disambiguatorOutput = pipeline.createDisambiguatorOutput(disambiguatorInput)

4: conllFormatOutput = pipeline.convertToConll(disambiguatorOutput)

5: nerInput = pipeline.createNerInput(disambiguatorOutput)

6: nerOutput = pipeline.createNerOutput(nerInput)

7: dependencyOutput = pipeline.createDependencyOutput(conllFormatOutput)

8: isQuantitative = pipeline.checkQuantitativeAnalysis(dependencyOutput)

9: **if** isQuantitative == TRUE **then**

10: sparqlComponents = QuantitativeAnswer.generateSparqlComponents(s)

11: final_query = QuantitativeAnswer. generateSparqlQuantitative (sparqlComponents, nerOutput)

12: **end if**

13: **else**

14: final_query = pipeline.generateSparql (disambiguatorOutput, nerOutput, dependencyOutput)

Further explanations for the methods mentioned in the Algorithm 1 are explained below to understand the algorithms better.

createMorphAnalyzerOutput: Method is utilized from the web service of ITU NLP tools. For a given sentence, morphologically analyzed tokens the sentence is returned as output. The output produced here is needed to be disambiguated.

createDisambiguatorInput: This method is used to reformat the morphologically analyzed output to apply disambiguation. Data format should be compatible with the format defined by the web service of ITU.

createDisambiguatorOutput: Method is utilized from the web service of ITU NLP tools. Output produced by ***createDisambiguatorInput*** method is the input parameter of this method. Disambiguated tokens are returned as the output.

convertToConll: Converts the output produced by ***createDisambiguatorOutput*** method to standard CoNLL format which is used by ***createDependencyOutput*** method.

createNerInput: This method is used to reformat the morphologically disambiguated

output to apply name entity recognition. Data format should be compatible with the format defined by the web service of ITU.

createNerOutput: Method is utilized from the web service of ITU NLP tools. Output produced by ***createNerInput*** method is the input parameter of this method. Recognized name entities are returned as output.

createDependencyOutput: Method is utilized from the web service of ITU NLP tools. Output produced by ***convertToConll*** method is the input parameter of this method. Dependency analysis of the sentence is returned as output.

checkQuantitativeAnalysis: Checks the sentence requires quantitative analysis or not, decides the sentence is Type 1 or Type 2. (See Section 3.3.2)

generateSparqlComponents: This method is used to build a Weka classifier that uses Multilayer Perceptron technique to classify sentence and generate the components which is utilized to formulate structured SPARQL query. (See Section 3.3.2)

generateSparqlQuantitative: The components generated by ***generateSparqlComponents*** method are used for the formulation of SPARQL query due to the aggregate function which is the answer type of the query. (See Section 3.3.2)

Flowchart of the Algorithm 1 is illustrated with Figure 3.2.

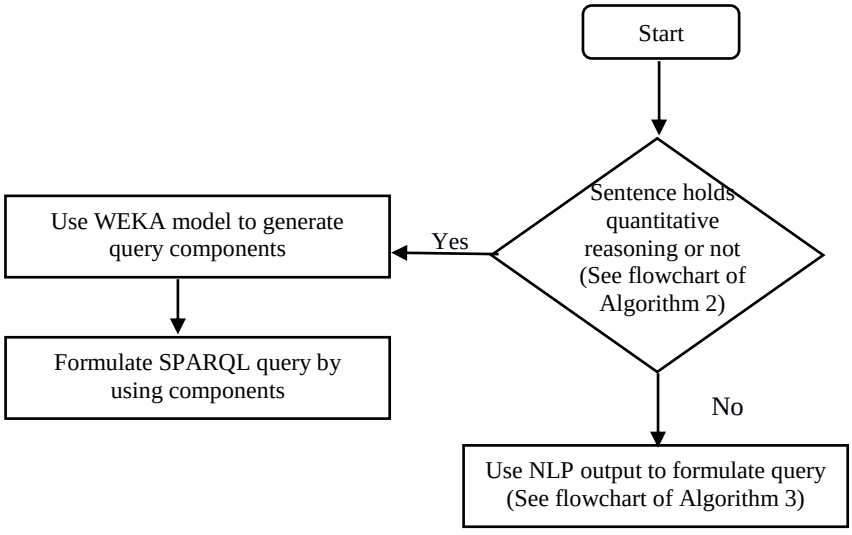


Figure 3.2. Flowchart of Algorithm 1

3.3. Query Formulation

3.3.1. Background Information: WEKA, Supervised Learning and Multilayer Perceptron

The Waikato Environment for Knowledge Analysis (WEKA) (Hall et al., 2009) which is a data mining software that provides researchers and practitioners easy access to state-of-the-art techniques in machine learning. Not only providing a platform to test the learning algorithms, but also a platform that can be customized by adding implementations of new algorithms without having any concern about supporting issues for data manipulation and scheme evaluation is also served by WEKA. Performing experiment and comparison on various machine learning algorithms on data sets is allowed. Data can be loaded from various sources, including files, URLs and databases. Supported file formats include WEKA's own ARFF format, CSV, LibSVM's format, and C4.5's format. Additionally, with a modular and extensible architecture, it is easy to use and integrate Weka facilities to your own implementation environment by using simple API. Regression, classification, clustering, association rule mining and attribute selection are included in the tool.

Supervised learning procedure is a type of "learning by example" instead of "learning by observation". Train data set is prepared to create learning model. Current input is tested by using learning model and predictions are obtained. Multilayer perceptron algorithm is used to predict the components of the given input sentence which is based on backpropagation algorithm which performs learning on a multilayer feed-forward neural network. Set of weights are iteratively learned for prediction of the categorical variables and it falls to the supervised neural networks (Arora, 2012). Input layer, one or more hidden layers and an output layer are the components of a multilayer perceptron network (See Figure 3.3).

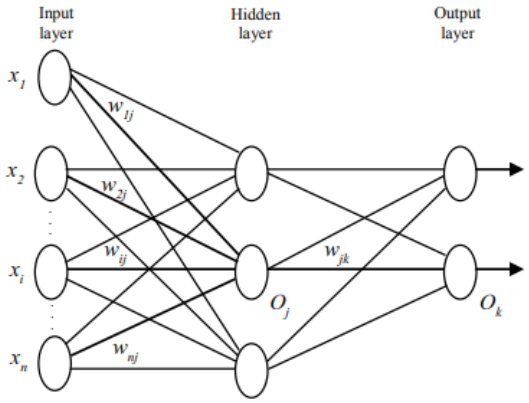


Figure 3.3. A multilayer feed-forward neural network (Arora, 2012)

Multilayer perceptron network has 3 parameters to be customized according to the nature and volume of data. Parameters should be optimized to find the best prediction. These parameters are learning rate, momentum and hidden layers (Han, Pei and Kamber, 2011).

Learning rate: Learning rate represents the degree of the training speed of the network. In other words, as the learning rate increases, the network trains faster but at the cost of the possibility of generating an unstable network.

Momentum: The momentum parameter is used to balance the network, prevents the problems possibly the caused by selecting a high learning rate which makes the network unstable.

Hidden layers: By adding the hidden layers, more target functions and combinations of the input features are represented (Minsky and Papert, 1988).

3.3.2. Answer and Question Type Detection and Query Formulation

Pipeline starts with question pre-processing with NLP techniques which is described in Section 3.2. Answer type detection is the second step to the target design and implementation of the question answering framework. Deciding on a type of the answer, discovering the mention in the question and understanding the user intent are critical tasks for a typical question answering system. This process continues with Algorithm 2 which is designed and implemented for deciding whether answer type includes quantitative reasoning or not and decides the question is Type 1 or Type 2. A rule based approach is applied for the method ***checkQuantitativeAnalysis.*** Requirement of quantitative reasoning analysis expression is checked on the tokens of the sentence by detecting the expressions like “kaç tane/kaç” (how many), “ne kadar”

(how many) or “en (superlative expression in Turkish - Adverb)”. After detecting the quantitative reasoning expression, algorithm further checks for the bigram of the tokens and try to detect “Adjective + Noun”, “Adverb + Adjective” or “Adverb + Noun” patterns.

Algorithm 2 Algorithm designed and implemented to check any expression holds quantitative reasoning (method name: ***checkQuantitativeAnalysis***)

Require: dependency analysis output

agenda: checks whether pre-processed sentence requires quantitative reasoning analysis or not

Ensure: isQuantitative

```
1:  isQuantitative == FALSE
2:  if dependencyOutput.contains (“kaç tane”) or (“kaç”) or (“ne kadar”) or (“en”) then
3:      if connectedToken.isAdjective() or  connectedToken.isAdverb()  then
4:          if next(connectedToken).isNoun() or  next(currentToken).isNoun() or
                currentToken.isAdjective()  then
5:          isQuantitative == TRUE
6:          end if
7:      end if
8: end if
```

If the patterns are found in the question which means that isQuantitative returns true, the question is determined as Type 2 and components of the query which are target class, entity class, data property, object property and function name are predicted by using supervised learning method. ARFF format is chosen from the valid file formats of WEKA to prepare train set for this thesis. Set of questions are trained and generated a train model to classify the test input. Multilayer perceptron which is an artificial neural network is employed for prediction. With using WEKA, predictions of the attributes defined in the learning model are added for natural language input by classification in the method ***generateSparqlComponents.*** Flowchart of Algorithm 2 is indicated in Figure 3.4.

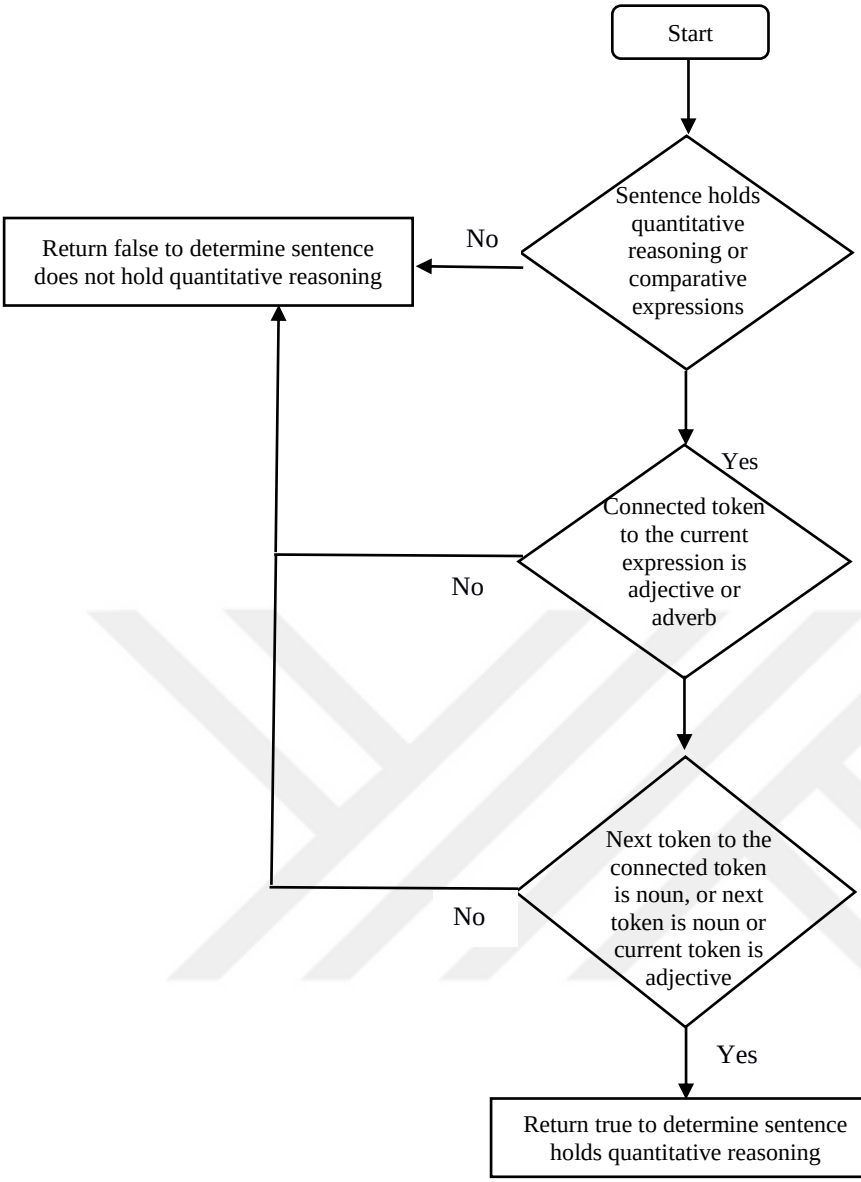


Figure 3.4. Flowchart of Algorithm 2

Multilayer perceptron is trained to predict the query components in a supervised manner. Train set has 1 string attribute and 5 nominal attributes which are also called categorical variables. String attribute is named *"sentence"*. Nominal attributes used in the train set are named *"target-class"*, *"entity-class"*, *"data-property"*, *"object-property"* and *"function_name"*. All categorical variables are modelled according to the context of the ontology which covers Chapter 6 of the geography lesson which is named "Spatial Synthesis: Turkey" in secondary school in 10th grade. Target class is the class in the ontology that defines the answer type of the sentence. Ontology class definition of a named entity in the sentence is represented with the entity class. Class names in the ontology are: *Şehir (City), Bölge (Region), Ülke (Country), Dağ*

(Mountain), Nehir (River), Göl (Lake), Ada (Island), Ova (Plain), Deniz (Sea) and İlçe (County). Possible data and object properties in the ontology are modelled as the candidates of the answer type in a sentence. Names of the data properties in the ontology are *yüzölçümü (surface area), popülasyon (population), yükseklik (height), derinlik (depth), tuzluluk (salinity), ortYağış (average rainfall), sıcaklık (temperature), enlemBoylam (longitude/latitude), bitkiÖrtüsü (vegetation), başkent (capital) and iklim (climate)*. Object property candidates are *konumlanır (locatedIn)*, *konumVar (hasLocations)* and *komsu (neighbourOf)*. Function name represents the name of the aggregate function handling the quantitative reasoning which is required to answer the question. Attributes and sample data from the train set are indicated below:

@attribute sentence string
@attribute target-class {Sehir,Bolge,Ulke,Dag,Nehir,Gol,Ada,Ova,Deniz, Ilce,null}
@attribute entity-class {Sehir,Bolge,Ulke,Dag,Nehir,Gol,Ada,Ova, Deniz, Ilce,null}
@attribute data-property
{yuzolcumu,populasyon,yukseklik,derinlik,tuzluluk,ortYagis,sicaklik, enlemBoylam, bitkiOrtusu,baskent,null,iklim}
@attribute object-property {konumlanir,konumVar,komsu,null}
@attribute function_name {count,min,max,sum,null}
@data
"Türkiye'nin en sığ denizi hangisidir",Deniz,Ulke,derinlik,konumlanir,min
(Which sea is the shallowest in Turkey?)
"Türkiye'nin en derin denizi hangisidir",Deniz,Ulke,derinlik,konumlanir,max
(Which sea is the deepest in Turkey?)
"Türkiye'de en fazla yağış alan il hangisidir",Sehir,Ulke,ortYagis,konumlanir,max
(Which city has the most rainfall in Turkey?)
"Marmara Bölgesi'nde kaç ada bulunur",Ada,Bolge,null,konumlanir,count
(How many islands does Marmara Region have?)
"Yıllık ortalama yağış miktarı en düşük ilimiz
hangisidir",Sehir,null,ortYagis,null,min
(Which city has the least average annual rainfall?)

Given sentence input is classified according to the sentence train model and results which are elicited as the components of the query. Output extracted from confusion

matrix shows that number of correctly classified instances 66 out of 100 questions. Algorithm in the method **generateSparqlQuantitative** decides how to formulate the query according to the type of function name predicted. Type of the aggregate function specifies whether to use subquery based pattern or not. For the functions min and max, subquery formation is inevitable because of the inherent of the Sparql queries, whereas count and sum functions do not require it. Subquery based pattern holds the same components but it differs on formation of the query. Structure of the subquery based pattern is showed below. Annotations for the prefixes (geo_turkce, ins) are extracted by using OWL API[5].

SELECT ?y ?min
WHERE { ?y rdf:type **ontology_name_prefix**:**target-class** .
?y **property_prefix**:**data-property** ?min .
{ SELECT (**function_name**(?var) as ?min)
WHERE { ?x rdf:type **ontology_name_prefix**:**entity-class** .
?y rdf:type **ontology_name_prefix**:**target-class** .
?y **property_prefix**:**object-property** ?x .
?y **property_prefix**:**data-property** ?var
FILTER(regex(str(?x),"**named entity**","i")) }
}} .

For instance; to answer the question: Türkiye'nin en sığ denizi hangisidir? (Which sea is the shallowest in Turkey?), SPARQL query that must be generated is:

SELECT ?y ?min
WHERE { ?y rdf:type **geo_turkce**:**Deniz** .
?y **ins**:**derinlik** ?min .
{ SELECT (**MIN**(?var) as ?min)
WHERE { ?x rdf:type **geo_turkce**:**Ulke** .
?y rdf:type **geo_turkce**:**Deniz** .
?y **ins**:**konumlanir** ?x .
?y **ins**:**derinlik** ?var
FILTER(regex(str(?x),"**Turkiye**","i")) }
}} .

[5] https://github.com/owlcs/owlapi/

As seen from the formulation same pattern is used for the similar type of questions whose function names are min or max which requires subquery involving. SPARQL components required to formulate the query can be listed as: *target-class* is Deniz (Sea), named entity is Türkiye so *entity-class* is Ulke (Country). *object-property* is the relation between target-class and named entity that holds user intention implicitly whose variable is *konumlanir (located)* for that case. In addition to these components, while generating all types of SPARQL queries, name of the ontology and prefixes are extracted by using the OWL API that reaches our ontology: GEO-TR. *data-property* is the property of the *target-class* but in the ontology, there is no property like "sığ" (shallow). There is only one valid property which is the antonymous of deepness (derinlik) that represents the amount of deepness for a sea. Therefore, quantitative analysis is required for the mentioned data property with the adverbial expression specified. "En sığ" stands for using the property "derinlik" with the aggregate function minimum. Instead of using traditional NLP techniques and any lexicon corpus, supervised learning [32] is applied to achieve semantics and for the framework to learn that type of expressions. Machine learning model contributes this question answering framework by training the query formulations to improve the precision, recall and F-measure metrics.

On the other hand, both for the aggregate functions count and sum, subquery based pattern is not required which makes query formulation easier and more comprehensible.

SELECT (**function_name** (?y) as ?total)
WHERE { ?x rdf:type **ontology_name_prefix**: **entity-class**.
?y rdf:type **ontology_name_prefix**: **target-class**.
?y **property_prefix**: **data-property** ?x
FILTER(regex(str(?x),"**named entity**","i")) }

For the sentence: "Marmara Bölgesi'nde kaç ada bulunur?" (How many islands does Marmara Region have?), *target-class* is Ada (Island), named entity is "Marmara", so *entity-class* is Bolge (Region). Required aggregate function is COUNT to count the islands which are located in the specified region. *object-property* is the relation between the named entity and the target class which is named as konumlanir (located) in the ontology. Therefore SPARQL components to formulate the query are: COUNT, Bolge, Ada, konumlanir and named entity Marmara.

SPARQL query for the sentence is:

SELECT (**COUNT** (?y) as ?total)
WHERE { ?x rdf:type **geo_turkce**:**Bolge** .
?y rdf:type **geo_turkce**:**Ada** .
?y **ins**:**konumlanir** ?x
FILTER(regex(str(?x),"**Marmara**","i")) }

For the condition that any quantitative reasoning analysis expression is not detected in the sentence (Type 1) by the algorithm (Algorithm 1 – Step 14), NLP techniques which are combined with ontology technologies are applied to represent the sentence with query expression. Algorithm 3 based on finding the answer type basically focuses on dependency analysis and name entity recognition output of the question. For Turkish language, user intent is mostly located on *OBJECT* or *SUBJECT* of a sentence or any directly dependent token to them which is the main assumption motivated by the rules of Turkish grammar while solving the problem. For this reason, pipeline proposes a solution that employs NLP output and to improve the accuracy. Ontology-supported techniques are also applied in this study. Sample processing steps for the question "Ankara iline komşu olan illeri gösterir misin?" (Can you show the neighbour cities of Ankara?). Dependency analysis for the current example is showed below.

1 Ankara Ankara Noun Noun Prop|A3sg|Pnon|Nom _ 2 POSSESSOR
2 iline il Noun Noun A3sg|P3sg|Dat _ 4 MODIFIER
3 komşu komşu Adj Adj _ _ 4 MODIFIER
4 olan ol Verb Verb Pos^DB|Adj|PresPart _ 6 MODIFIER
5 illeri il Noun Noun A3pl|Pnon|Acc _ 6 OBJECT
6 gösterir göster Verb Verb Pos|Aor|A3sg _ 7 ARGUMENT
7 misin mi Postp Postp Ques|Pres|A2sg _ 0 PREDICATE
8 ? ? Punc Punc _ _ 7 PUNCTUATION

Dependency analysis result showed the type of token roles in the sentence. Token 5 is

the object of that sentence which will play a critical role for answer type detection . Axiom type of the object is checked to decide on whether it is a class, a data or object property or an individual. For the sample case, axiom type for token 5 (“il” (city)) is a class in our ontology and accepted as the target class for query generation. After detecting the axiom type of the object phrase, algorithm decides on the action types. If it is a class (Algorithm 3 – Step 5), then the properties of that class with the named entity (if exists) in the sentence is found to generate SPARQL query. Finding named entity is critical to achieve the properties in the query so named entity recognizer result is utilized. Named entity result of this sentence is showed below.

Ankara Ankara+Noun+Prop+A3sg+Pnon+Nom B-LOCATION

iline il+Noun+A3sg+P3sg+Dat O

komşu komşu+Adj O

olan ol+Verb+Pos^DB+Adj+PresPart O

illeri il+Noun+A3pl+Pnon+Acc O

gösterir göster+Verb+Pos+Aor+A3sg O

misin mi+Postp+Ques+Pres+A2sg O

? ?+Punc O

“Ankara” is the named entity for this sentence. Entity class for the individual “Ankara” is extracted from the ontology as Sehir (entity class) which is same class with the object phase. Words “il” (target class) and “sehir” are synonyms in Turkish which both mean city. Possible relations with “Ankara” and class Sehir are extracted from the ontology. Only relation is found out to be an object property “komsu” (nameOfproperty) for that sample case. Generic pattern for SPARQL formulation in Algorithm 3 is:

SELECT ?y

WHERE { ?x rdf:type **ontology_name_prefix**: **entity-class**.

?y rdf:type **ontology_name_prefix**: **target-class**.

?y **property_prefix**: **nameOfproperty** ?x

FILTER(regex(str(?x),"**named entity**","i")) }

By using this pattern, query is generated as the following:

SELECT ?y
WHERE { ?x rdf:type **geo_turkce**:**Sehir** .
?y rdf:type **geo_turkce**:**Sehir** .
?y **ins**:**komsu** ?x .
FILTER(regex(str(?x),"**Ankara**","i")) }

In order to better understand the Algorithm 3, second sample question that includes a

subject phrase is represented. For the question “Ege Bölgesi’nin yüzölçümü ne kadardır? (How much is the land area of Aegean region?)”, dependency analysis and named entity recognition results are given below.

Dependency analysis result:

1 Ege ege Noun Noun A3sg|Pnon|Nom _ 2 POSSESSOR

2 Bölgesi’nin bölge Noun Noun A3sg|P3sg|Gen _ 3 POSSESSOR

3 yüzölçümü yüzölçüm Noun Noun A3sg|P3sg|Nom _ 5 SUBJECT

4 ne ne Pron Pron Ques|A3sg|Pnon|Nom _ 5 ARGUMENT

5 kadardır kadar Postp Postp PCNom^DB|Noun|Zero|A3sg|Pnon|Nom^DB|Verb|Zero|Pres|A3sg|Cop _ 0 PREDICATE

6 ? ? Punc Punc _ _ 5 PUNCTUATION

Named entity recognition result:

Ege ege+Noun+A3sg+Pnon+Nom B-LOCATION

Bölgesi'nin bölge+Noun+A3sg+P3sg+Gen I-LOCATION

yüzölçümü yüzölçüm+Noun+A3sg+P3sg+Nom O

ne ne+Pron+Ques+A3sg+Pnon+Nom O

kadardır kadar+Postp+PCNom^DB+Noun+Zero+A3sg+Pnon+Nom^DB+Verb+Zero+Pres+A3sg+Cop O

? ?+Punc O

Token 3 is the subject of the question that represents the answer type. Axiom type is checked for the stemmed form of the token 3. Answer type of the subject expression is found out to be a data property which is also a sign for the requirement of quantitative analysis. Answer type should be “yüzölçümü” (surface area) but first, classes in the sentence which have this data property should be detected to formulate the query (See Algorithm 3 – Step 31). From NER result, “Ege Bölgesi” (Aegean Region) is the named entity and entity class is extracted as “Bölge (Region)” from ontology. Utilizing from dependencies between words provide the related token with token 3. Token 2 which is directly dependent to token 3. So, axiom type of this related expression (“Bölge”) is checked from the ontology (Step 21). Results show that axiom type is a class. So, algorithm is moving back to Step 22. First thing is to find the properties but for that case answer type is a property itself so searching for the common

connected with the subject expression is the second thing to do (Step 25). Common connected is already found because of the fact that there is no other entity for that case and system formulates the query.

SELECT ?variable
WHERE { ?x rdf:type **geo_turkce**:**Bolge** .
?x **ins**:**yuzolcumu** ?variable .
FILTER(regex(str(?x),"**Ege**","i")) }

Final sample case is a slightly more complex one and contains a determinative group for possessive construction ("zincirleme isim tamlaması") to demonstrate how Algorithm 3 handles that type of sentences. For the question: "Ege Bölgesi'ndeki şehirlerin nüfuslarını gösterir misin ? (Can you show me the populations of the cities in Aegean Region?)", dependency analysis and NER result are given below.

Dependency analysis result:

1 Ege ege Noun Noun A3sg|Pnon|Nom _ 2 POSSESSOR
2 Bölgesi'ndeki bölge Noun Noun A3sg|P3sg|Loc^DB|Adj|Rel _ 5 MODIFIER
3 şehirlerin şehir Noun Noun A3pl|Pnon|Gen _ 4 POSSESSOR
4 nüfuslarını nüfus Noun Noun A3pl|P3sg|Acc_ 5 OBJECT
5 gösterir göster Verb Verb Pos|Aor|A3sg _ 6 ARGUMENT
6 misin mi Postp Postp Ques|Pres|A2sg _ 0 PREDICATE
7 ? ? Punc Punc _ _ 6 PUNCTUATION

Named entity recognition result:

Ege ege+Noun+A3sg+Pnon+Nom B-LOCATION
Bölgesi'ndeki bölge+Noun+A3sg+P3sg+Loc^DB+Adj+Rel I-LOCATION
şehirlerin şehir+Noun+A3pl+Pnon+Gen O
nüfuslarını nüfus+Noun+A3pl+P3sg+Acc O
gösterir göster+Verb+Pos+Aor+A3sg O
misin mi+Postp+Ques+Pres+A2sg O
? ?+Punc O

After detecting the token 4 ("nüfuslarını" (population)) is the object of the sample sentence so algorithm at first checks the axiom type for the stemmed form ("nüfus") of object expression (Algorithm 3-Step 4). "Nüfus" is the synonym for the data

property which is named “populasyon” in the ontology. The function checkAxiomType returns the data property as result for the input “nüfus”. Algorithm 3 continue to run with Step 10 by calling the findRelatedToken method which returns the directly dependent token to the token 4. Token 3 (“şehirlerin”) has a dependency relation with token 4. Dependency analysis is critical here, not to find the population of Aegean region. Required answer should be the population of cities in Aegean region. Stemmed form of token 3 “şehir” (city) is checked for the axiom type from the ontology and result returns back as class that moves the algorithm to Step 6 by assigning the answerType to the related token (“şehir”). Properties defined between “Ege Bölgesi” and “şehir” is extracted from the ontology and only one object property returns which is “konumVar (hasLocation)”. Named entity, entity class, target class, data and object properties are assigned to formulate the query.

SELECT ?variable
WHERE { ?x rdf:type **geo_turkce**:**Sehir** .
?y rdf:type **geo_turkce**:Bolge .
?y **ins**:**konumVar** ?x .
?x **ins**:**populasyon** ?variable .
FILTER(regex(str(?y),"**Ege**","i")) }

Algorithm 3 Algorithm to find the answer type of question in Turkish and generate SPARQL query by using processed output by NLP techniques (Method name: generateSparql)

Require: sentence processed by NLP techniques
agenda: generate query by using NLP output
Ensure: final_query
1: nerEntities = pipeline.getNamedEntities(nerOutput)
2: **if** *dependencyOutput.contains(“OBJECT”)*
3: answerType = objectTerm
4: axiomType = pipeline.checkAxiomType(answerType)
5: **if** *axiomType == “CLASS”* **then**
6: properties = pipeline.findProperties(answerType, nerEntities)
7: *final_query* = pipeline.formulate_query(properties, nerEntities, answerType)

8: **end if**

9: **if** *axiomType == "DATA PROPERTY"* **then**

10: relatedToken = pipeline.findRelatedToken(answerType, dependencyOutput)

11: axiomTypeRelated = pipeline.checkAxiomType(relatedToken)

12: Go back to Step 5 call the method for the input axiomTypeRelated and continue again

13: **end if**

14: **if** *axiomType == "OBJECT PROPERTY"* **then**

15: relatedClass = pipeline.findRelatedToken(answerType,dependencyOutput)

16: *final_query* = pipeline.formulate_query(answerType, nerEntities, relatedClass)

17: **end if**

19: **if** *dependencyOutput.contains("SUBJECT")* **then**

20: answerType = subjectTerm

21: axiomType = pipeline.checkAxiomType(answerType)

22: **if** *axiomType == "CLASS"* **then**

23: properties = pipeline.findProperties(answerType, nerEntities)

24: **if** *properties == NONE* **then**

25: commonConnected = pipeline.findCommonConnected(dependencyOutput, answerType)

26: Go to Step 21 call the method for the input commonConnected and continue again

27: **end if**

28: **end if**

29: **else**

30: *final_query* = pipeline.formulate_query(properties, nerEntities, answerType)

31: **if** *axiomType == "DATA PROPERTY"* **then**

32: relatedToken = pipeline.findRelatedToken(answerType, dependencyOutput)

33: axiomTypeRelated = pipeline.checkAxiomType(relatedToken)

34: Go back to Step 21 call the method for the input axiomTypeRelated and

continue again

35: **end if**

36: **if** *axiomType == "INDIVIDUAL"* **then**

37: connectedToken = pipeline.findConnectedToken (answerType, dependencyOutput)

38: axiomTypeConnected = pipeline. checkAxiomType(connectedToken)

39: Go back to Step 21 call the method for the input axiomTypeConnected and cont. again

40: **end if**

41: **if** *axiomType == "OBJECT PROPERTY"* **then**

42: commonConnected = pipeline.findCommonConnected(dependencyOutput, answerType)

43: Go back to Step 21 call checkAxiomType for the input commonConnected and continue

44: **end if**

45: **end if**

Further explanations for the methods mentioned in the Algorithm 3 and the flowchart of Algorithm 3 are illustrated on Figure 3.5 to understand the algorithm better.

checkAxiomType: For a given input term, this method checks the existence of the axiom from the given ontology and extracts the type of the axiom if exists. If not exists, method uses a Weka classifier to find the most-nearer expression from the defined axioms in the ontology and then decides on the type of the axiom.

findProperties: For the given named entities and token which is found as subject or object in a sentence or any other term connected to them, this method finds the defined properties in the ontology to use them while generating the query.

formulate_query: For the given named entities, properties and token which is a target object token which is found as subject or object in a sentence or any other term connected to them, are formulated to generate a structured SPARQL query.

findRelatedToken: This method finds the token which is directly dependent to the token which is found as subject or object in a sentence or any other term connected to them.

findCommonConnected: For a given token which is found as subject or object in a sentence or any other term connected to them, findCommonConnected returns the token which depends also the same token with them. Finding the token which is

dependent commonly by the input parameters.

findConnectedToken: This method gets the name of the token which subject or object in a sentence or any other term connected to them is directly dependent.

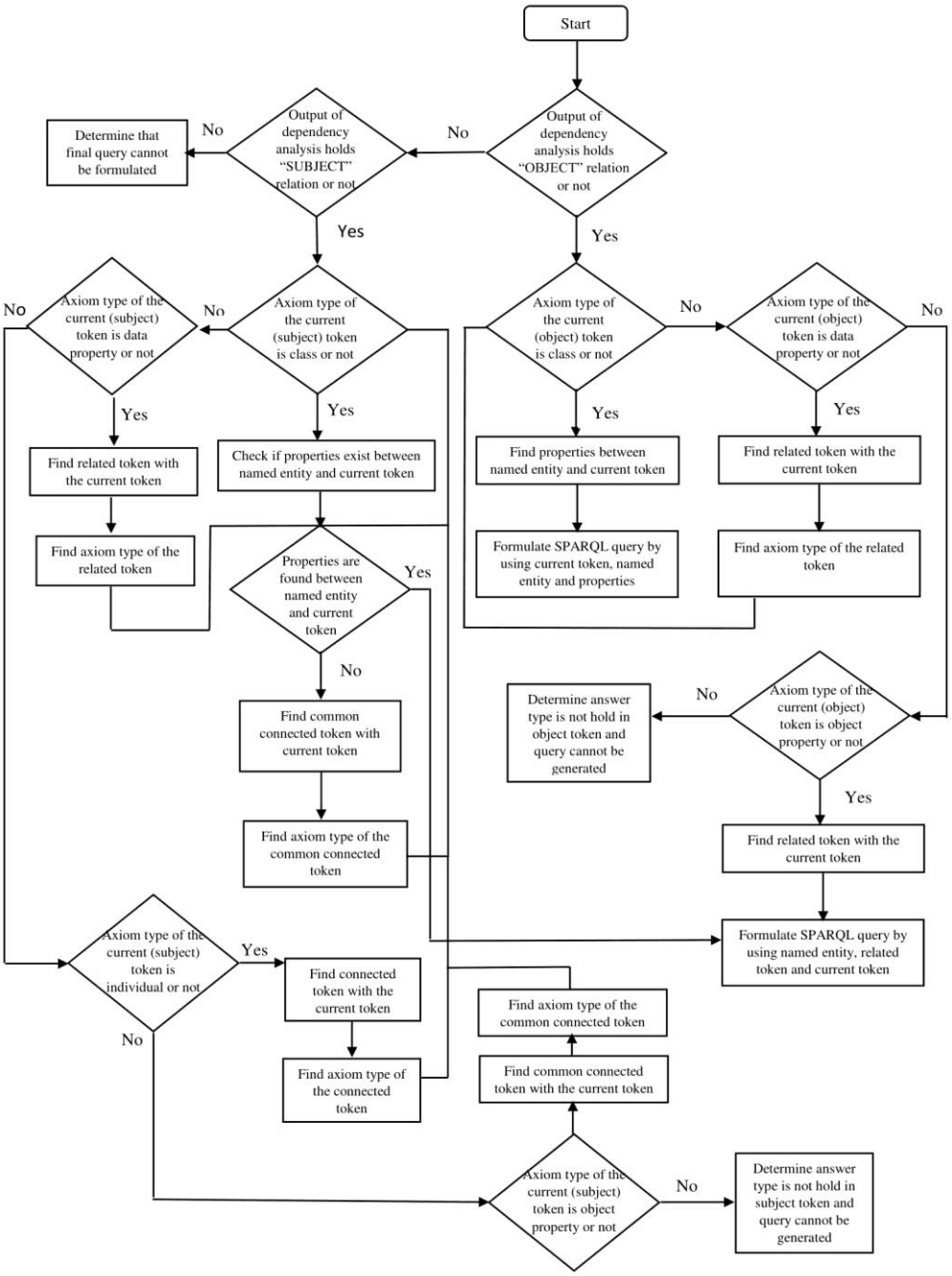


Figure 3.5. Flowchart of Algorithm 3

3.4. Ontology Development

3.4.1. Definition of Ontology

Ontology as a word roots back to Greek splitting in two parts as ontos for "being" and logos for "word". Philosophy science defines it as subject of existence (Gasevic, Djuric and Devedžic, 2006) and categorizing the existence is valid in some domain (Sowa, 2000). Domain ontology formally representing the categories of things for a specific domain like a conceptual model. For computer science and information science, in ontologies knowledge is represented with set of concepts and relations between those concepts. By means of this, sharing a common vocabulary which models a specific domain with the definitions of concepts, their properties and relations are served by ontologies structurally.

Gruber (1993) defined ontology as "formal, explicit specification of a shared conceptualization". Two expressions should be focused on to better comprehend this definition. First one is conceptualization which is an abstract, simplified view of the world for some specific purpose. According to Gruber's definition (1993) of conceptualization, every knowledge-based system or knowledge-level agent is committed to some conceptualization, explicitly or implicitly (Gruber, 1995). Every activities even in daily life as ways of behaving, every form of represented knowledge is based on a certain conceptualization. Conceptualization consists of system of concepts which represents some phenomenon in which is composed of objects, processed and relations in various sorts of ways (Smith, 2003). Second word is to be concentrated on is specification which has means of defining the ontology with a declarative, explicit and formal representation. Explicit and formal specification refers for an ontology to be machine readable which implies that declaratively representing the knowledge that holds in it (Gasevic, Djuric and Devedžic, 2006).

Another definition from Hendler (2001) that clarifies ontology as a set of knowledge terms which is composed of vocabulary, semantic interconnections and some simple rules of inference and logic for some particular domain. The most significant part of this definition is the semantic interconnections, inferencing and logic other than the Gruber's definition. Hendler points out that conceptually related words in terms of semantic definitions are interconnected by using the logical rules and inferencing mechanism. In other words, ontologies enable various forms of reasoning.

According to Kalfoglou (2001), ontology is an explicit representation of a shared understanding of the important concepts in some domain of interest. Definition of Kalfoglou (2001), emphasizes the shared understanding to prevent the problems of subjectivity. Objectivity is represented as an agreement about subjectivity and explicit cognitive structure is employed to achieve that goal. That feature of ontologies provides the facility to share and reuse knowledge which results in semantic interoperability between intelligent agents and applications.

General description for an ontology is formal explicit specification of concepts in a domain of discourse. Discourse refers to the classes and various features and attributes of a discourse refers to properties. Set of instances of classes are called individuals which creates a knowledge base. Classes are really critical while developing an ontology to form conceptualization in a declarative and accurate way. Creating subclasses that are more specific than classes is also possible while developing ontologies.

3.4.2. Motivation for Ontology Development

Main motivation for developing an ontology is the requirement of sharing information in a specific domain and providing basic concepts and relations which are machine understandable. Additionally, with the linked data concept semantic interoperability, knowledge sharing and reusing is served by ontologies. Main reasons can be listed as follows: sharing common knowledge and understanding, reusing of domain knowledge, explicitly declaring domain assumptions and analyzing the domain knowledge.

Sharing common knowledge and understanding: Extracting and integrating information and sharing the same knowledge from different sources is a key aspect for standardization. Computer agents or systems can communicate by using the same language which addresses the same underlying ontology of terms. Musen (1992) and Gruber (1993) express that common understanding of information structure among people or software systems is enabled. Combined information is used to answer user queries or as input data to other systems (Noy and McGuinness, 2001).

Reusing of domain knowledge: Reusability principle fits in ontologies easily. Similar domain studies simply focus on reusing domain knowledge that is represented in ontologies. Integration of one or more ontologies on a specific domain is also possible to build large ontologies. In addition to this, an existing ontology can be reused, saved

for customization or extension.

Explicitly declaring domain assumptions: If the knowledge about the domain changes, assumptions about the domain can be changed easily. Managing the domain assumptions easily also helping the hard-coding assumptions about the implementation phase even for someone without programming knowledge. Meaning of the terms in specific domain for novice users is handled by explicit specifications of domain knowledge.

Analyzing the domain knowledge: Analysis of the explicitly declared specification of the terms is made analyzing the domain knowledge possible. While extending and reusing the ontologies, domain knowledge analysis and analysis of terms are beneficial.

Ontologies are not executable items, they require defining a set of data and their structure for other applications and systems to use. Data is generated from ontologies by problem solving methods, domain-independent applications and software agents.

Main motivation for this thesis, developing an ontology which has a geographic domain for only the spatial information of Turkey. GEO-TR holds the information about the geographical details of Turkey which described with scope of the ontology and competency questions in Step 1 for ontology development. GEO-TR ontology is compatible with employing with a Turkish question answering framework which is represented in this study. All data and object properties, individuals and class names are in Turkish. Instead of using an another ontology which is designed in English or in any language like Geonames[6] ontology. A decision is taken to develop an ontology in Turkish that holds spatial data of Turkey. The main reason of this decision to prevent extra efforts while performing translation between languages. Literature gap is discovered in the point that literature is lack of a Turkish ontology in geographic domain. In addition to this, properties, relations and classes did not meet the requirements of the system objectives of this thesis.

While designing the ontology and question answering framework to compatible each other, the most considered principles are flexibility, adaptability, modifiability and portability for our question answering framework to be compatible with other ontologies in different domains. Making possible to extend the ontology and reuse the knowledge represented is also taken into consideration.

[6] http://www. geonames.org/ontology

Developing a new ontology generally involves the following practical steps:

- class definitions,
- systematically arranging the class hierarchy,
- preparing taxonomy,
- filling the individuals and arranging the classes,
- defining properties and allowed values for them,
- assigning values for properties by mapping with the individuals of the ontology.

Steps of ontology development 101 (Noy and McGuinness, 2001) is followed as a guideline to develop GEO-TR. Activities performed during development are mapped each of the steps in the guideline.

Step 1: Determine the domain and scope of the ontology

First step of ontology development is drawing the boundaries by determining the domain and scope of the ontology. During that phase questions that should be answered can be listed as:

- Which type of domain ontology will cover?
- For what the ontology will be used?
- What types of questions will be answered by ontology?

GEO-TR is representing the spatial data about Turkey. Geographic domain is covered. To meet the requirement of limiting the scope of the model, specific usage of the ontology should be discussed. GEO-TR is designed for the usage of the students in secondary school in 10^{th} grade. Chapter 6 of the geography lesson which is named "Spatial Synthesis: Turkey" is the target scope. The subsections of this chapter can be listed as:

Characteristics and distribution of landforms

Climate of Turkey

Vegetation of Turkey

Features of geospatial model of Turkey

Water resources of Turkey (rivers, seas, lakes).

Types of the questions that can be asked to the ontology is the set of competency questions. Defining competency questions is a way of drawing a sketch of a list of questions that can be answered by the ontology (Grüninger and Fox, 1995). They are just a sketch to represent the fundamental issues in ontology, they do not need to be

sophisticated. These questions are helpful to validate and check the completeness of the ontology additionally. Main concerns about writing competency questions to find the answers to the questions such as:

"Does the ontology contain enough information to answer these types of questions?"

"Do the answers require a particular level of detail or representation of a particular area?".

Sample competency questions are listed below.

- *Türkiye'deki şehirleri gösterir misin?*
  (Can you show the cities in Turkey?)
- *İzmir'in komşularını gösterir misin?*
  (Can you show the neighbours of Izmir?)
- *Antalya ili hangi coğrafi bölgededir?*
  (Which geographical region is Antalya located in?)
- *Türkiye'de bulunan coğrafi bölgeleri gösterir misin?*
  (Can you show the geographical regions in Turkey?)
- *Akdeniz bölgesinde bulunan dağları gösterir misin?*
  (Can you show the mountains in Mediterranean region?)
- *Manisa şehrinin çevresinde hangi şehirler konumlanır?*
  (Which cities are located in neighbourhood of Manisa?)
- *Türkiye' de hangi iklim kuşakları bulunmaktadır?*
  (Which climatic zones situated in Turkey?)
- *Ege Bölgesi'nin toplam nüfusu ne kadardır?*
  (What is the total population of Aegean region?)
- *İzmir' in en yüksek dağı hangisidir?*
  (Which is the highest mountain in Izmir?)
- *Türkiye' de en fazla yağış alan il hangisidir?*
  (Which city has the most rainfall in Turkey?)
- *Türkiye' nin nüfus yoğunluğu en yüksek ili hangisidir?*
  (Which city has the highest population density in Turkey?)
- *Yüzölçümü en küçük coğrafi bölgemiz hangisidir?*
  (Which geographical region has the smallest surface area?)

- *Elazığ ilinde kaç tane ilçe bulunur?*
  (How many districts are there in Elazığ?)

Step 2: Consider reusing existing ontologies

Reusing existing ontologies are considered carefully for this study, but the problems about the language of the ontology, mapping ontology concepts with the components of the analyzed sentence and customized knowledge requirement of this framework directed this study to develop an ontology from scratch.

Step 3: Enumerate important terms in the ontology

For this step, important terms which refer the concepts of the ontology, the points that we want to make statements and represent knowledge. Determining the main elements with conceptualization provides defining the scope, boundaries and hierarchies easier. After declaring these concepts, properties of the elements and possible relations between them are declared. This step prevents overlapping between concepts they represent, relations among terms or any properties they might have. Creating a basis for design phase is performed and it is critical for further ontology development activities.

Important concepts in GEO-TR are determined as Ada ("Island"), Bogaz (Strait), Bolge (Region), Dag (Mountain), Deniz (Sea), Gol (Lake), Nehir (River), Ova (Plain), Sehir (City), Ilce (District) (subclass of Sehir) and Ulke (Country). Table 3.4 shows the important concepts and related terms for GEO-TR.

Table 3.4. Important concepts and related terms in the domain

Important Concepts	**Related Terms**
Ada	konumlanir (locatedIn), nufus (population),
Bogaz	konumlanir (locatedIn), uzunluk (length)
Bolge	konumlanir (locatedIn), konumVar (hasLocations), nufus (population), yuzolcumu (surface area)
Dag	konumlanir (locatedIn), yukseklik (height)
Deniz	konumlanir (locatedIn), derinlik (depth), tuzluluk (salinity)
Gol	konumlanir (locatedIn), derinlik (depth)
Nehir	konumlanir (locatedIn), uzunluk (length)
Ova	konumlanir (locatedIn), yuzolcumu (surface area)
Sehir	konumlanir (locatedIn), konumVar (hasLocations), nufus (population), yuzolcumu (surface area), yukseklik (height), ortalamaYagis (average rainfall), komsu (neighbourOf)
Ilce	konumlanir (locatedIn), nufus (population), yuzolcumu (surface area)
Ulke	konumlanir (locatedIn), konumVar (hasLocations), nufus (population), yuzolcumu (surface area), iklim (climate), baskent (capital)

Step 4: Define the classes and the class hierarchy

Several approaches are possible while developing a class hierarchy (Uschold and

Gruninger, 1996). Well known and most popular approaches are top-down, bottom-up and combination approaches.

Top-down approach: Firstly, defining the most general concepts in the specified domain and specializing for lower levels. Direction followed during design goes from top (more general concepts) to the down (more specialized concepts).

Bottom-up approach: Defining the most specialized concepts at first, then grouping is performed on those concepts to find more general definitions, concepts to represent knowledge.

Combination approach: Main definition is combining the top-down and bottom-up approaches. Most remarkable concepts are defined and rules for the generalization and specialization are decided as the model grows. While designing top level and low level concepts, sometimes middle level concepts are inserted for properly assigning the hierarchy for knowledge representation.

None of these three approaches are better than the others. Approach selection sharpens by the inherent of a domain and personal view. Depending on the systematic view of the ontology developer, top-down or bottom-up or combination approaches are utilized. Middle level concepts can be more descriptive in the domain therefore most ontology developers decide to choose the combination approach (Rosch, 1978). Also for GEO-TR, combination approach is followed both for designing the level of concepts and it should be noted that because of the inherent of the geographical domain, developing class hierarchy is rarely used. Concepts are different levels and no hierarchy involved between them except the classes “Thing” and all other classes and “Sehir” and “Ilce”. Classes are extracted from the important terms specified in Step 4. Ada (“Island”), Bogaz (Strait), Bolge (Region), Dag (Mountain), Deniz (Sea), Gol (Lake), Nehir (River), Ova (Plain), Sehir (City), Ilce (District) (and Ulke (Country) are determined as the classes of GEO-TR. List of classes and class hierarchy are illustrated on Figure 3.6 and Figure 3.7.

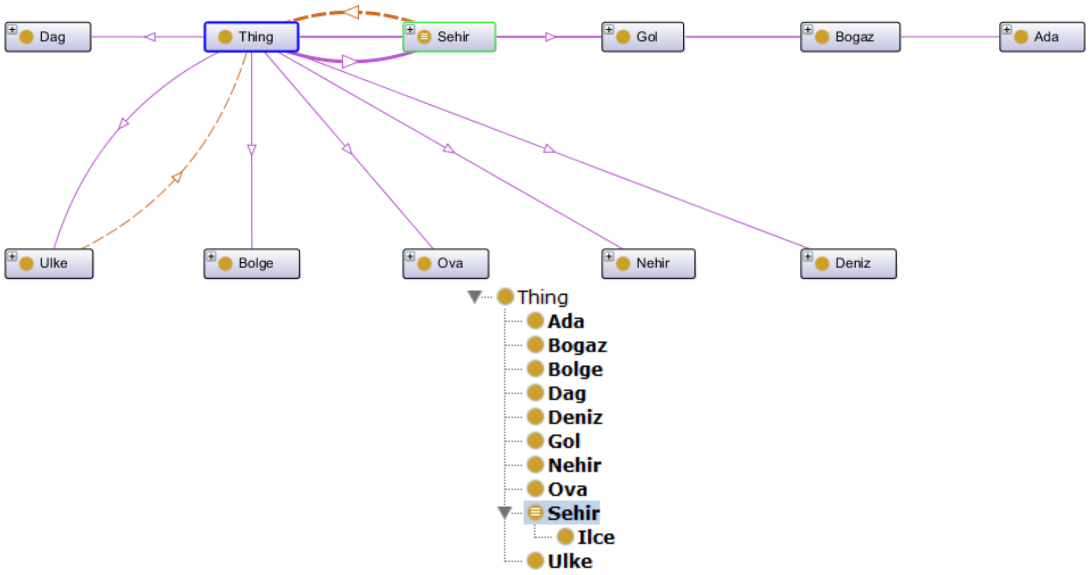


Figure 3.6. List of classes in GEO-TR – OntoGraf view Protégé

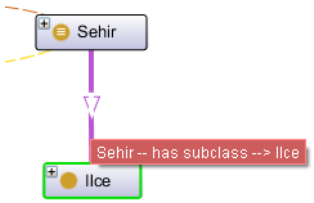


Figure 3.7. Class hierachy between Sehir and Ilce – OntoGraf view Protégé

Step 5: Define the properties of classes - slots

By enumerating the important terms in the domain, classes of the ontology is already selected. Representation of classes are not enough to answer the competency questions that is described in Step 1. Features of the specified important terms is required. In Step 3, related terms are described which are the related concepts with the classes. These related terms represent the properties of classes. For example, a mountain has a property height, a river has a property locatedIn, length, sea class has a property salinity, city class has a property locatedIn, neighbourOf, etc. Referencing the Table 3.4, properties of the classes are extracted and displayed on Figure 3.8 and Figure 3.9.

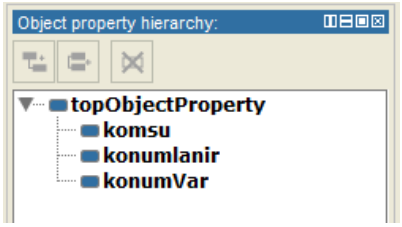


Figure 3.8. Object properties in GEO-TR

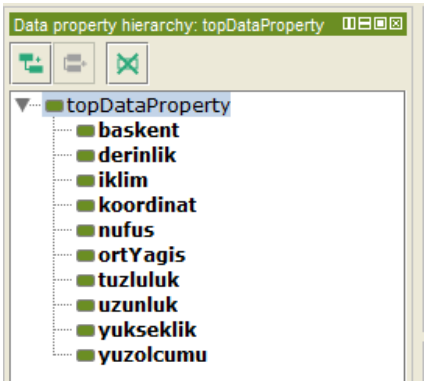


Figure 3.9. Data properties in GEO-TR

Class definition is determined for each property in the list in other means, describing which property is valid in which class. There are two types of properties in an ontology, first one is object properties and the second is data properties. Main principle to follow is that object properties link individuals to individuals whereas data properties link individuals to data values, this type of a relation holds attribute in it.

Domain of the ontology is specified for geography. So, most fundamental relation between concepts is determined as "konumlanir (located)" which is an object property. "konumVar" is the inverse property of "konumlanir". For better understanding this relation, sample individual list and example with individuals are demonstrated in Step 7: creating instances.

Characteristics of object properties should be decided while designing the relations between concepts and instances. These characteristics are addressing functional features of the properties and possible characteristics provided by Protégé are demonstrated in Figure 3.10.

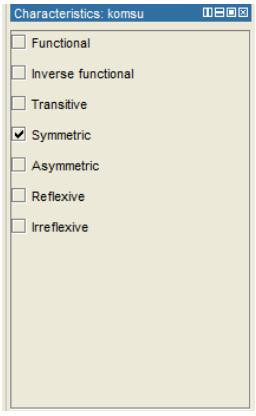


Figure 3.10. Characteristics of an object property

Functional: For any given individual, if the specified object property can have at most one individual, then it is asserted that selected property is functional. To explain it in

another way, at most one outgoing relation is defined for that individual along this property. In addition to this Functional characteristics is the only feature for data properties as can be seen in Figure 3.11.

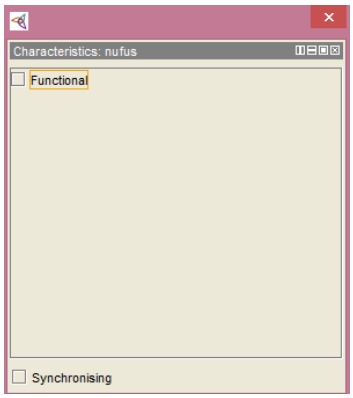


Figure 3.11. Characteristics of a data property

Inverse Functional: Inverse property of the specified property is Functional which means that there can be at most one ingoing relation via this property for that individual.

Transitive: This characteristic asserts that selected property is transitive between the individuals. Transitive property implies that if individual x has a relation with y, y has a relation with z then, x has the same relation with z.

Symmetric: For the given individuals x and y, if x has a symmetric relation with y than y must have the same relation with individual x along that property. If a property is symmetric, then the property should also be defined inversely. For example: "komsu" (neighbourOf) is a symmetric property in GEO-TR which implies that if a city x has neighbourOf relation with another city y, then city y also must have neighbourOf relation with city x.

Asymmetric: Asymmetric characteristic asserts that if an individual x has an asymmetric relation with individual y along an object property, then the inverse of that relation cannot be defined via the same property.

Reflexive: Every single individual which has reflexive property, each of them are related to itself along that property.

Irreflexive: Every single individual which has irreflexive property, each of them cannot be related to itself along that property.

Step 6: Define the facets of properties - slots

Facets of a property can describe value type, allowed values, cardinality (number of

allowed values) and other features of the values a property can have. For example: “nüfus” (population) (data property) of a city should be an integer (see Figure yy). Another example for an object property can be given for “konumlanir” (locatedIn) like “a City is located in a Country” or “a City is located in a Region”. “konumlanir” can have more than one value and also from the instances of different classes. Another variant of this example is the object property which is named “konumVar”.

Slot cardinality

Slot cardinality describes the number of values a property can have. Possible restriction type values for property cardinality are some, only (universal), minimum, exactly and maximum. Some represents existential restrictions. To be more specific, set of individuals are allowed to have at least one relation to individuals that are members of a class in the ontology.

Existential restrictions describe the set of individuals that have at least one specific kind of relation to individuals from a specific class. Only represents universal restrictions are also known as single cardinality. Minimum cardinality is described to allow minimum number of values for specified property can have. Exactly is to allow exact and precise number of values along the defined property. Last restriction type is maximum cardinality which defines maximum number for the values while defining a property. All restriction types are shown in Figure 3.12.

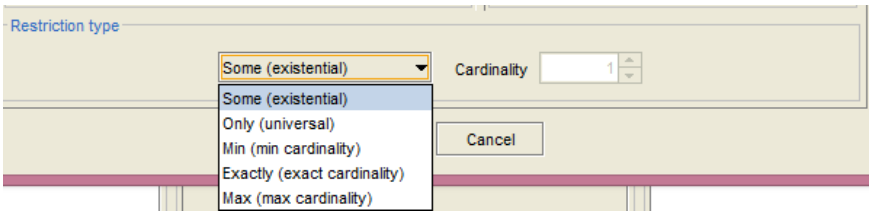


Figure 3.12. Restriction types in cardinality

Example for a single cardinality can be given as a city has only one “population” value whereas multiple cardinality for the property “neighbourOf” for a city can have more than one value. Another example a Country can have exactly 1 value for the relation “capital”. Giving minimum or maximum restriction types are not applied for the properties in GEO-TR.

Slot-value type

Value type of a slot or property describes the type of values a property or relation can

hold in it. Common value types are String, Number, Boolean, Enumerated and Instance.

String: Value of the property is restricted as a simple String. For example, “komsu” (neighbourOf) property has holds the name of the neighbour which is a string.

Number: Value of the property holds numerical data which can have specific types like double, float, integer etc. For example, “nufüs” (population) property holds a numerical value which has integer type.

Boolean: Boolean restriction type simply represents yes/no flags. Any boolean restriction is not used in GEO-TR.

Enumerated: List of possible allowed values are defined as a list in enumerated properties. Any enumerated relation is not used in GEO-TR.

Instance: Instance typed properties define the relation between individuals. Additionally, a list of possible classes which may have relation with the specified attributes can also be defined. For example, instance of a class “Sehir” (city) has “komsu” (neighbourOf) relation with another instance of “Sehir” class.

Domain and range of a property - slot

Range of an object property describes the allowed class type of instances which can have the specified property. For example as illustrated in Figure 3.13, class “Sehir”, “Ulke” and “Bolge” can be selected to define the ranges of “komsu” property. Instances of these classes are valid to have “komsu” (neighbourOf) property in GEO-TR.

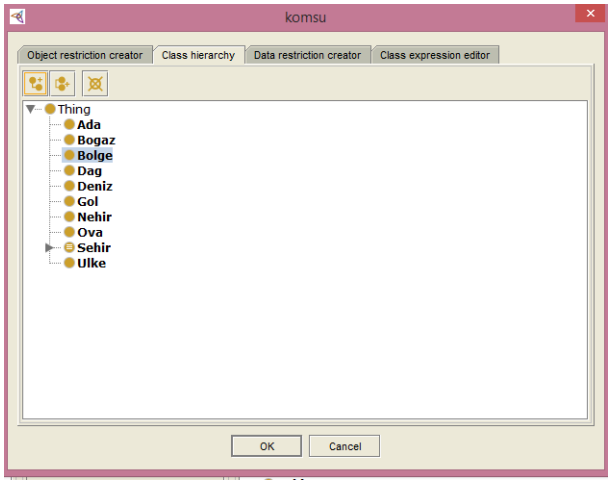


Figure 3.13. Defining range for an object property

Range of a data property is defining the data type restrictions. For example, string is

the range of “baskent” (capital) property (see Figure 3.14). A capital city cannot be defined with any other type than string in GEO-TR.

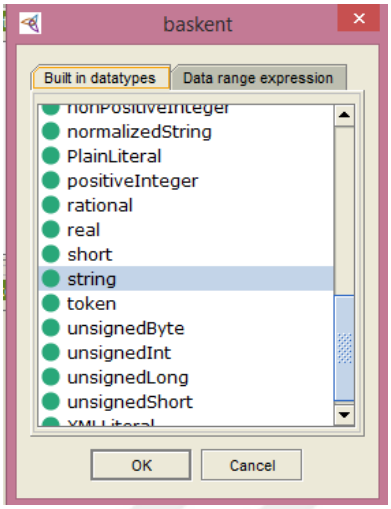


Figure 3.14. Defining restriction for data property

Domain of an object property describes the attached classes to a property. The classes which the property is attached generate the domain of that property so no need to specify the domain separately which will result in redundant information. For example, the “Sehir” (city) class is the domain of the located in slot. Another example is demonstrated on Figure 3.15.

Figure 3.15. Object property “konumlanir” between the classes “Ulke” and “Sehir” and “Sehir” and “Ilce”

Fundamental rules while determining range and domain of the properties follow the similar principles. Firstly, the most general class or classes are found. All the classes in the domain or range of a property should be described by the property and individuals of that classes should be the potential fillers for that property.

Step 7: Create instances

Final step is creating individuals for designed class hierarchy and defined properties. Simple procedure to follow has 3 steps. First one deciding on a class type and then creating an individual instance of the selected class. Final step is to fill the property values which the newly created individual has. Sample list of individuals from various

classes, class “Sehir” (city) and “Bolge”(region) can be seen in Figure 3.16, Figure 3.17 and Figure 3.18.

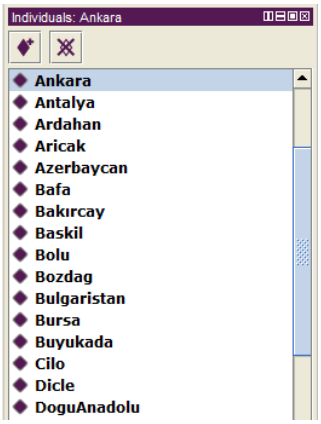


Figure 3.16 Sample list of individuals

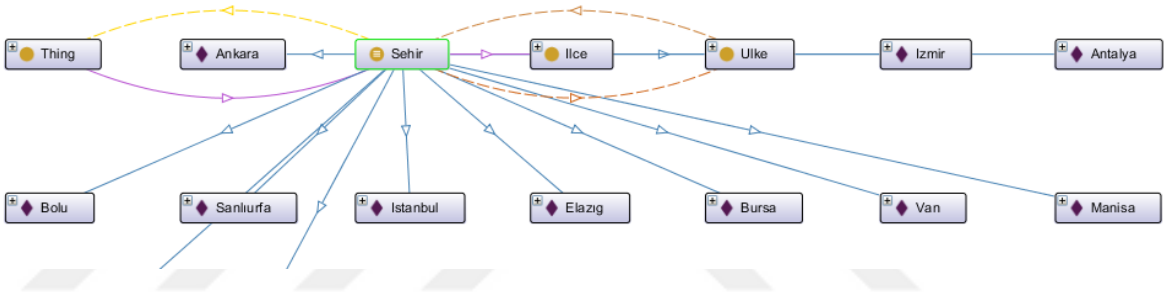


Figure 3.17. Sample list of individuals of “Sehir” class

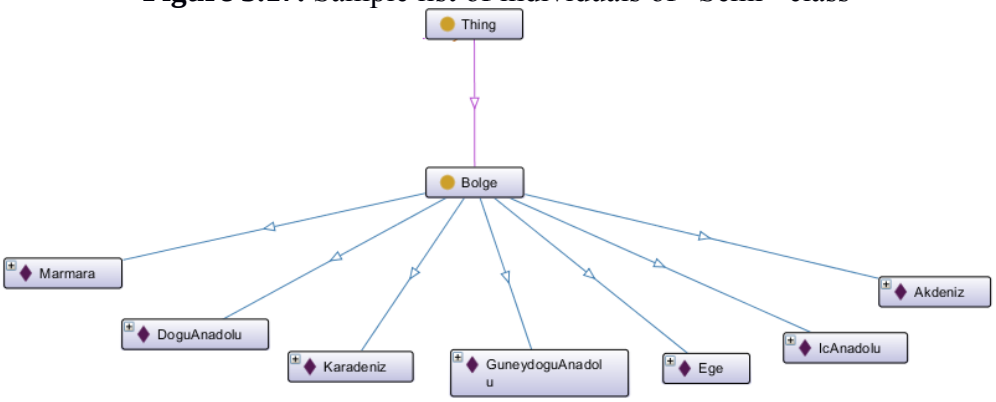


Figure 3.18. Sample list of individuals of “Bolge” class

In Figure 3.19, another example relation “konumlanir” is demonstrated between the individuals “Ankara (instance of class Sehir-City)” and “Turkiye (instance of class Ulke-Country)”.

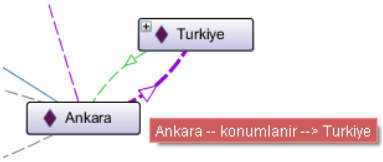


Figure 3.19. Object property: "konumlanir" between individuals "Ankara" and "Turkiye"

For example, for the individual "Manisa", fullfilled object properties are "konumlanir" (locatedIn) and "komsu" (neighbourOf) and data properties are "ortYagis"(average rainfall), "yukseklik"(height), "nufus"(population) and "yuzolcumu" (surface area) (See Figure 3.20).

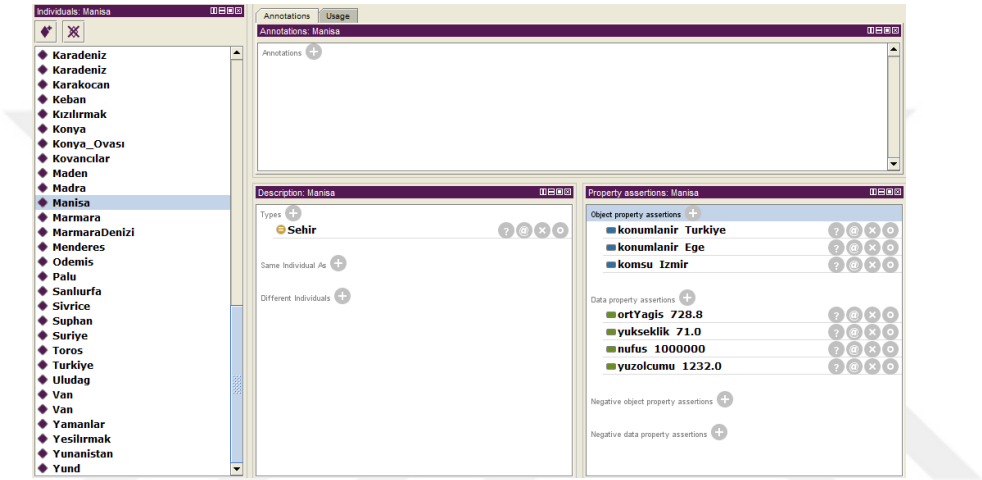


Figure 3.20. Details arranged for individual "Manisa"

3.5. Experimental Study and Comparison

Question answering framework represented in this thesis, handles two types of questions (Type 1 and Type 2) mentioned in Section 3.2. Test data set used for the comparison consists of only Type 1 questions. The questions that require quantitative reasoning (Type 2) are answered by using machine learning. Learning model generated by supervised learning method is used to predict the query components. Predicted components are target class, entity class, data property, object property and aggregate function name to perform quantitative reasoning analysis for the given attributes. Predicted components are used to formulate the SPARQL query. Comparing this method with two types of questions defined in Section 3.2 is unreasonable. Because, Type 2 questions are processed with learning mechanism. Only two types of methods are used as comparing paradigms. First one is hybrid approach that combines NLP based and ontology based approach and the second method is using ontology-based approach alone. Both of these methods are not applied to Type 2 questions, so they are

not involved in the test data set used for comparison. Therefore, test data set only consists of informative questions which are named as Type 1 in this study.

Questions are answered by using two types of methods which are focusing on different aspects from the same dimension. First approach is using both NLP output and ontology technologies. Second one is using entirely ontology based methodology. NLP with ontology-supported solution is the solution of combining NLP output with the ontology technologies for an effort of finding the accurate entities and the relations between them to construct triples to generate SPARQL query. This method is based on a double-checking model. Double checking model mentions the checking NLP output for the algorithm to decide and checking again the NLP output and the relations from the ontology whether they exist or they have that kinds of relations or not. Whereas the second method is only based on ontology checking which means that tokenizing every word in the question and checking the existence and possible relations between them from the ontology. Generating the SPARQL query with the components that are found as output of checking the defined ontology is the motivation for the second method.

A basic visual interface is implemented for experimental study. Figure 3.21 shows the screenshot of the interface.

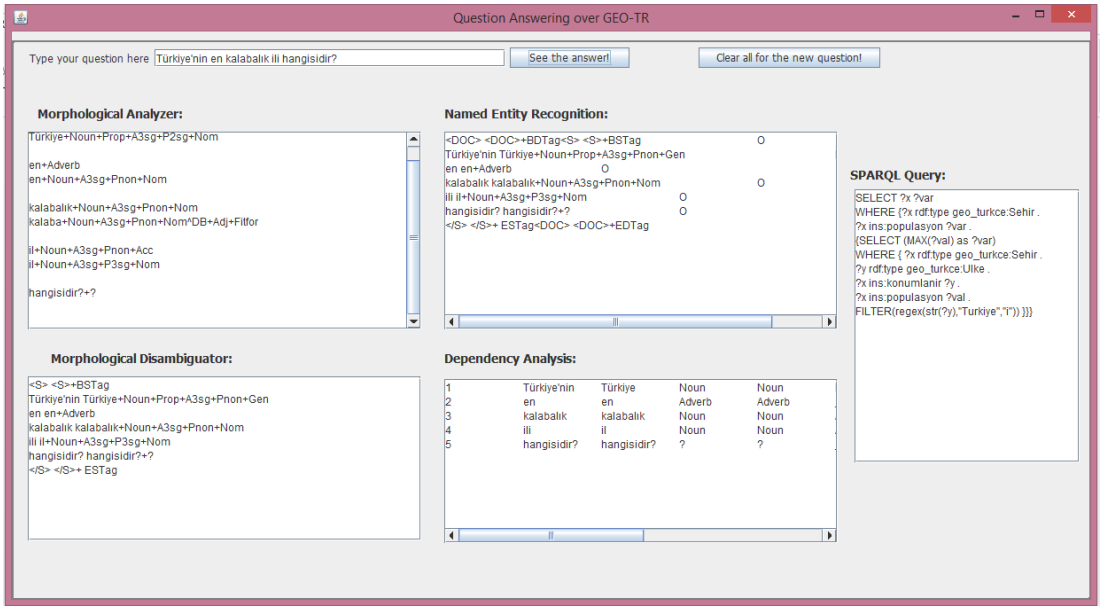


Figure 3.21. User interface for experimental study

Experimental study is performed to compare these two methods defined. Experiment is conducted with 100 questions that are listed in Appendix 1. Metrics for comparison are precision, recall and F-measure. Precision, recall and F-measure can be

summarized as performance metrics of the systems in terms of quality. Precision is the fraction of retrieved answers that are relevant to a query. All retrieved results are taken into account. Whereas for recall, only relevant or standard answers are considered while computing. Lastly, F-measure is the harmonic mean of precision and recall. Formulations for three of them are given in Section 2.3.6. Results of the experimental study is given Table 3.5.

Table 3.5. Comparison of Method 1 and Method 2

Method	**Precision**	**Recall**	**F-Measure**
Method 1: Hybrid approach	0.77	0.68	0.71
Method 2: Ontology based approach	0.64	0.57	0.60

Results show that combination of NLP techniques with ontology-based solutions, improves the results other than using ontology based approach alone. Morphologically disambiguated forms of words and relationships between words have great contribution to achieve the meaning. Questions that holds possessive construction are good examples that require dependency analysis to extract the related tokens to decide on accurate answer type. Method 1 utilizes dependency analysis and generates accurate answer whereas ontology based method directly fetches the semantic items, relations between words are not analyzed in Method 2 which results in generating inaccurate answer. A good example that shows that dependency analysis is definitely required while processing sentences which hold possessive construction is the sample sentence: "Ege Bölgesi'ndeki şehirlerin nüfuslarını gösterir misin? (Can you show the populations of cities in Aegean Region?)". Ontology based method cannot disambiguate whether the populations of the each city in Aegean Region is intended by the user or the population of the Aegean Region which is an individual in GEO-TR. Population is a data property in the ontology which is defined both for cities and regions. The relation POSSESSOR between the tokens "şehirlerin" and "nüfuslarını" disambiguates the intended question is asking the population of the cities which are located in Aegean Region.

Experimental results show that hybrid approach has better results in terms of precision, recall and F-measure. Each method has its own advantages and drawbacks but considering the overall performance, Method 1 which has hybrid approach that combines NLP-based and ontology-based techniques generate more accurate results.

District names are not annotated during named entity recognition step in hybrid approach. They are not captured so answer cannot be generated. On the other hand, ontology based approach directly captures distinct names as individuals belongs to the District class. These questions are answered by Method 2. For a question like “Sivrice ilçesi hangi bölgedir? (In which region is Sivrice district located?)”, Method 1 cannot generate an answer but Method 2 extracts “Sivrice” as an individual of District class and “bölge (region)” as class name Region from ontology. Finally, algorithm finds the object property between “Sivrice” and Region class as “konumlanir (locatedIn)” and formulates the accurate query. Another sample question: “Ödemiş’te bulunan ovaların yüzölçümlerini sıralar mısın? (Can you list the land areas of plains in Ödemis?”) which has possessive construction (“zincirleme isim tamlaması”). It results in inaccurate answer from both methods. Method 1 fails because of the mentioned fact above. On the other hand, dependency analysis which is critical for the expressions that has possessive construction is not involved in Method 2. Using only the ontology to answer this question is not adequate and results in generation of inaccurate answer. Method 2 cannot disambiguate the user intention is on the land area of Odemis or plains located in Odemis without knowing the relations between words. “ovaların” and “yüzölçümlerini” tokens have POSSESSOR relation which disambiguates land areas of plains is intended by the user.

Method 2 simply checks each token exists in ontology or not, possible relations and types are detected if they found in the ontology. Simple rules are used to decide on answer type and generate triple for SPARQL queries. A simple rule can be defined as class type found other than entity class type is a candidate for answer type. A drawback of ontology based method is another point to discuss for the sentences which are involved of more than one class, Method 2 cannot disambiguate the answer type. A good simple example can be given as: “İzmir şehri hangi bölgededir?”. After processing the sentence, query components are listed as İzmir, Sehir and “Bolge”. Possible relations between individual and classes that are “konumlanir” (Izmir-Bolge) and “komsu”(Izmir-Sehir) are extracted. Question can be answered by showing the region of neighbour cities of Izmir.

CHAPTER 4
CONCLUSION AND FUTURE WORKS

Main aim of this thesis study is to design and implement a geographic Turkish question answering framework over linked data (GEO-TR). Literature gap in Turkish question answering systems which utilize linked data in geographical domain is addressed and described as the main motivation for this study. Main motivation, scope and contributions of the study with background information are given in Chapter 1. Concepts related question answering over linked data are defined to better comprehend the methodologies and techniques of the study. Literatur gap is addressed to motivate on the contribution of this thesis study. Literature review is given with a systematic research method in Chapter 2. State of art techniques are discussed and compared with evaluation paradigms. The description of main methodology for development of geographic question answering over linked data (GEO-TR) is presented in Chapter 3. Steps and algorithms involved in the development phase are explained in detail. In addition to this, linked data provider which is named GEO-TR; a novel Turkish ontology on this area is described. The development of this new Turkish ontology is accepted as one of the main contribution of this thesis. The other contribution is to develop a Turkish question answering framework in geographic domain. Following the rules of ontology development 101, steps of GEO-TR ontology development are represented. Later on, an experimental study is conducted with 100 questions which corresponds to the competency questions derived from geography lessons in 10^{th} grade secondary school curriculum. Comparison is given between two methods which are hybrid approach that combines NLP based techniques with linked data technologies and ontology based techniques. Results show that hybrid approach which is applied on this thesis study has better performance in terms of precision, recall and F-measure values.

Types of queries which cannot be answered in this framework are detected after experimental study. More complex queries which have more than one recognized entity, more than one level possessive constructions or conditional and comparing expressions cannot be handled in the current version of thesis proposed. Handling these types of queries can be offered for future work and for the other researchers which are interested in this study area. To deal with this type of queries, a method which combines supervised learning, NLP techniques and linked data technologies can

be applied. Not only applying learning to Type 2 (sentences that requires quantitative reasoning) sentences, but also Type 1 sentences (informative sentences) can be processed by using learning procedures. Results extracted from learning can be combined with NLP results and ontology based techniques. A good example can be given as "Türkiye'nin yüzölçümü en büyük ikinci şehri hangisidir? (Which is second largest city of Turkey by land area?)". While generating answer of this sample question, quantitative reasoning is required to find largest city of Turkey. But this sentence also holds a conditional expression which filters with the expression second biggest city which filters the result. Conditional expressions can be caught by NLP techniques and query is generated by using the aggregate functions on linked data query language. Another example can be given from the sentence "Nüfusu 15 milyonun altındaki şehirleri göster (Show the cities of which population is under 15 million)" that holds conditional expression in it. Named entity recognition techniques can be combined with learning procedures to handle conditional expressions. Numeric expressions can be recognized with NER and the system can be trained with patterns of the conditional expressions. Additionally, instead of classifying the sentences with algorithms, learning algorithms or neural networks can be applied to classify sentences in different types or structures.

One of the main contribution of this study architecture proposed for this framework is designed to be flexible and applicable for the other domains. By using customized predefined categories for named entity recognition and ontologies from multiple different domains, multi-ontology supported platform can be provided. Algorithm generates SPARQL query is not dependent on the type of the domain. Domain plays role only for the ontology type and usage. Method used to convert natural language question to SPARQL query is free from the features of a specific domain which is believed to be a significant contribution to the literature.

This framework is based on only geographic domain, extending the coverage of domains and creating a multi-ontology platform can also be drawn as a future study direction. A pipeline that accepts a natural language input and classifies the sentence according to the domain types can be proposed to fit with this architecture. After determining the domain, corresponding ontology can be used as knowledge source.

An additional point to discuss for the future work is the deep learning techniques which gain increasingly attention for natural language related technologies and question answering systems. Techniques proposed with deep learning methods can entirely alter

the architectures and methods utilize for this research field and improve the quality metrics gained from the experimental study proposed in this thesis study.

REFERENCES

Kumar, E. (2011). *Natural language processing*. IK International Pvt Ltd.

Hoque, M. M., Rahman, M. J., & KumarDhar, P. (2007, March). Lexical Semantics: A Newapproach to Analyze the Bangla Sentence with Semantic Features. In *Information and Communication Technology, 2007. ICICT'07. International Conference on* (pp. 87-91). IEEE.

Hayes, P. J., & Carbonell, J. G. (1983). A tutorial on techniques and applications for natural language processing.

Guo, Y., Li, Y., & Shao, Z. (2011, May). A semantic processing model for sentence understanding based on cognitive learning. In *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on* (pp. 106-110). IEEE.

Temizer, M., & Diri, B. (2012, July). Automatic Subject-Object-Verb relation extraction. In *Innovations in Intelligent Systems and Applications (INISTA), 2012 International Symposium on* (pp. 1-4). IEEE.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*(Aug), 2493-2537.

Indurkhya, N., & Damerau, F. J. (Eds.). (2010). *Handbook of natural language processing* (Vol. 2). CRC Press.

Nivre, J. (2010). Dependency Parsing. Language and Linguistics Compass, 4(3), 138–152.

Shehata, S., Karray, F., & Kamel, M. (2007, November). Enhancing search engine quality using concept-based text retrieval. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 26-32). IEEE Computer Society.

Celikkaya, G., Torunoglu, D., & Eryigit, G. (2013, October). Named entity recognition on real data: a preliminary investigation for Turkish. In *Application of Information and Communication Technologies (AICT), 2013 7th International Conference on* (pp. 1-5). IEEE.

Jurafsky, D., & Martin, J. H. (2014). *Speech and language processing* (Vol. 3). London:: Pearson.

Zelle, J. M., & Mooney, R. J. (1996, August). Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*(pp. 1050-1055).

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific american*, *284*(5), 34-43.

Unger, C., Freitas, A., & Cimiano, P. (2014, September). An introduction to question answering over linked data. In *Reasoning Web International Summer School* (pp. 100-140). Springer, Cham.

Lopez, V., Uren, V., Sabou, M., & Motta, E. (2011). Is question answering fit for the semantic web?: a survey. *Semantic Web*, *2*(2), 125-155.

Lopez, V., Unger, C., Cimiano, P., & Motta, E. (2013). Evaluating question answering over linked data. *Web Semantics: Science, Services and Agents on the World Wide Web*, *21*, 3-13.

Höffner, K., Walter, S., Marx, E., Usbeck, R., Lehmann, J., & Ngonga Ngomo, A. C. (2017). Survey on challenges of question answering in the semantic web. *Semantic Web*, *8*(6), 895-920.

Walter, S., Unger, C., Cimiano, P., & Bär, D. (2012, November). Evaluation of a layered approach to question answering over linked data. In *International Semantic Web Conference* (pp. 362-374). Springer, Berlin, Heidelberg.

Noy, N. F., & McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology.

Guo, R., & Ren, F. (2009, September). Towards the relationship between Semantic Web and NLP. In *Natural Language Processing and Knowledge Engineering, 2009. NLP-KE 2009. International Conference on* (pp. 1-8). IEEE.

Pugsee, J., Evens, M. W., & Rivepiboon, W. (2006, July). Analysis of Thai Sentences with a Serial Verb Using a Semantic Lexicon. In *Cognitive Informatics, 2006. ICCI 2006. 5th IEEE International Conference on* (Vol. 1, pp. 484-494). IEEE.

Bernstein, A., Kaufmann, E., Göhring, A., & Kiefer, C. (2005, November). Querying ontologies: A controlled english interface for end-users. In *International Semantic Web Conference* (pp. 112-126). Springer, Berlin, Heidelberg.

Kitchenham, B., Charters, S., 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering. EBSE Technical Report, Keele and Durham University.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., & Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of systems and software*, *80*(4), 571-583.

Sackett, D. L. (1997). *Evidence-based Medicine How to practice and teach EBM*. WB Saunders Company.

Unger, C., Forascu, C., Lopez, V., Ngomo, A. C. N., Cabrio, E., Cimiano, P., & Walter, S. (2014, September). Question answering over linked data (QALD-4). In *Working Notes for CLEF 2014 Conference*.

Unger, C., Forascu, C., Lopez, V., Ngomo, A. C. N., Cabrio, E., Cimiano, P., & Walter, S. (2015, September). Question answering over linked data (QALD-5). In *Working Notes* for CLEF 2015 Conference.

Walter, S., Unger, C., & Cimiano, P. (2014). ATOLL—A framework for the automatic induction of ontology lexica. *Data & Knowledge Engineering*, *94*, 148-162.

Allam, A. M. N., & Haggag, M. H. (2012). The question answering systems: A survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS)*, *2*(3).

Xu, K., Feng, Y., & Zhao, D. (2014, September). Xser@ qald-4: Answering natural language questions via phrasal semantic parsing. In *Working Notes for CLEF 2014 Conference* (pp. 15-18).

Dima, C. (2014, September). Answering Natural Language Questions with Intui3. In *CLEF (Working Notes)* (pp. 1201-1211).

He, S., Zhang, Y., Liu, K., & Zhao, J. (2014, September). CASIA@ V2: A MLN-based Question Answering System over Linked Data. In *CLEF (Working Notes)* (pp. 1249-1259).

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine learning*, *62*(1-2), 107-136.

Park, S., Kwon, S., Kim, B., & Lee, G. G. (2015). ISOFT at QALD-5: Hybrid Question Answering System over Linked Data and Text Data. In *CLEF (Working Notes)*.

Hamon, T., Grabar, N., Mougin, F., & Thiessard, F. (2014). Description of the POMELO System for the Task 2 of QALD-2014. In *CLEF (Working Notes)* (pp. 1212-1223).

Usbeck, R., & Ngomo, A. C. N. (2015). HAWK@ QALD5-Trying to Answer Hybrid Questions with Various Simple Ranking Techniques. In *CLEF (Working Notes)*.

Speck, R., & Ngomo, A. C. N. (2014, October). Ensemble learning for named entity recognition. In *International semantic web conference* (pp. 519-534). Springer, Cham.

Ruseti, S., Mirea, A., Rebedea, T., & Trausan-Matu, S. (2015). QAnswer-Enhanced Entity Matching for Question Answering over Linked Data. In *CLEF (Working Notes)*.

Dolan-Gavitt, B., Leek, T., Zhivich, M., Giffin, J., & Lee, W. (2011, May). Virtuoso: Narrowing the semantic gap in virtual machine introspection. In *Security and*

Privacy (SP), 2011 IEEE Symposium on (pp. 297-312). IEEE.

C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, D. McClosky, The Stanford CoreNLP Natural Language

Beaumont, R., Grau, B., & Ligozat, A. L. (2015). SemGraphQA@ QALD5: LIMSI participation at QALD5@ CLEF. In *CLEF (Working Notes)*.

Baudiš, P., & Šedivý, J. (2015). Biomedical question answering using the yodaqa system: Prototype notes.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., & Hellmann, S. (2009). DBpedia-A crystallization point for the Web of Data. *Web Semantics: science, services and agents on the world wide web*, *7*(3), 154-165.

Cai, Q., & Yates, A. (2013). Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Vol. 1, pp. 423-433).

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111-3119).

Fader, A., Soderland, S., & Etzioni, O. (2011, July). Identifying relations for open information extraction. In *Proceedings of the conference on empirical methods in natural language processing*(pp. 1535-1545). Association for Computational Linguistics.

Nakashole, N., Weikum, G., & Suchanek, F. (2012, July). PATTY: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (pp. 1135-1145). Association for Computational Linguistics.

Choi, J. D., & Palmer, M. (2011, June). Getting the most out of transition-based dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2* (pp. 687-692). Association for Computational Linguistics.

Morton, T., Kottmann, J., Baldridge, J., & Bierner, G. (2005). Opennlp: A java-based nlp toolkit. EACL.

Yosef, M. A., Hoffart, J., Bordino, I., Spaniol, M., & Weikum, G. (2011). Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proceedings of the VLDB Endowment*, *4*(12), 1450-1453.

Schmid, H. (2013, November). Probabilistic part-ofispeech tagging using decision trees. In *New methods in language processing* (p. 154).

Aubin, S., Deriviere, J., Hamon, T., Nazarenko, A., Poibeau, T., & Weissenbacher, D. (2006). A robust linguistic infrastructure for efficient web content analysis: the ALVIS project. In *Proceedings of the Symposium on Digital Semantic Content across Cultures, Paris*.

Erguvanli, E. E., & Taylan, E. E. (1984). *The function of word order in Turkish grammar* (Vol. 106). Univ of California Press.

Eryiğit, G. (2014). ITU Turkish NLP web service. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 1-4).

Sahin, M., Sulubacak, U., & Eryigit, G. (2013, October). Redefinition of Turkish morphology using flag diacritics. In *Proceedings of The Tenth Symposium on Natural Language Processing (SNLP-2013), Phuket, Thailand, October*.

Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Şeker, G. A., & Eryiğit, G. (2012). Initial explorations on using CRFs for Turkish named entity recognition. *Proceedings of COLING 2012*, 2459-2474.

Şeker, G. A., & Eryiğit, G. (2017). Extending a CRF-based named entity recognition model for Turkish well formed text and user generated content 1. *Semantic Web*, *8*(5), 625-642.

Eryigit, G. (2012). The Impact of Automatic Morphological Analysis & Disambiguation on Dependency Parsing of Turkish. In *LREC* (pp. 1960-1965).

Eryiğit, G., Nivre, J., & Oflazer, K. (2008). Dependency parsing of Turkish. Computational Linguistics, 34(3), 357-389.

Buchholz, S., & Marsi, E. (2006, June). CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning* (pp. 149-164). Association for Computational Linguistics.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., ... & Marsi, E. (2007). MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, *13*(2), 95-135.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*, *11*(1), 10-18.

Arora, R. (2012). Comparative analysis of classification algorithms on different datasets using WEKA. *International Journal of Computer Applications*, *54*(13).

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Minsky, M., & Papert, S. (1988). *Perceptron* (expanded edition) Cambridge, MA, MIT Press (original edition: 1969).

Gaševic, D., Djuric, D., & Devedžic, V. (2006). *Model driven architecture and ontology development*. Springer Science & Business Media.

Sowa, J. F. (2000, August). Ontology, metadata, and semiotics. In *International Conference on Conceptual Structures* (pp. 55-81). Springer, Berlin, Heidelberg.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, *5*(2), 199-220.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing?. *International journal of human-computer studies*, *43*(5-6), 907-928.

Smith, B. (2003). Ontology. *Blackwell guide to the philosophy of computing and information*, Blackwell, Oxford, UK, 155–166.

Hendler, J. (2001). Agents and the semantic web. *IEEE Intelligent systems*, *16*(2), 30-37.

Kalfoglou, Y. (2001). Exploring ontologies. In *Handbook of Software Engineering and Knowledge Engineering: Volume I: Fundamentals* (pp. 863-887).

Musen, M. A. (1992). Dimensions of knowledge sharing and reuse. *Computers and biomedical research*, *25*(5), 435-467.

Grüninger, M., & Fox, M. S. (1995). The role of competency questions in enterprise engineering. In *Benchmarking—Theory and Practice* (pp. 22-31). Springer, Boston, MA.

Uschold, M., & Gruninger, M. (1996). Ontologies: Principles, methods and applications. *The knowledge engineering review*, *11*(2), 93-136.

Rosch, E. 1978: Principles of categorization. In: Rosch, E., Lloyd, B.B. (eds), *Cognition and categorization*. Hillsdale, NJ: Erlbaum. 27‑48.

APPENDIX 1 – Test Dataset Questions

1- Türkiye'de hangi şehirler vardır ?

2- Türkiye'de hangi kentler vardır ?

3- Türkiye'deki şehirleri gösterir misin ?

4- İzmir hangi ülkededir?

5- İzmir hangi bölgededir?

6- İzmir'in komşularını gösterir misin?

7- İzmir hangi coğrafi bölgededir ?

8- İstanbul şehri hangi bölgededir?

9- Antalya ili hangi coğrafi bölgededir ?

10- Ege bölgesinde hangi şehirler vardır ?

11- Akdeniz bölgesindeki şehirleri listeler misin ?

12- Türkiye'de bulunan coğrafi bölgeleri gösterir misin ?

13- Türkiye'deki coğrafi bölgeleri gösterir misin ?

14- Türkiye'deki coğrafi bölgeleri sıralar mısın ?

15- Akdeniz bölgesinde bulunan dağları gösterir misin ?

16- Yeşilırmak nehri hangi bölgededir ?

17- Ege Bölgesi'ndeki şehirlerin nüfuslarını gösterir misin ?

18- Ege Bölgesi'nin yüzölçümü ne kadardır?

19- Manisa şehrinin çevresinde hangi şehirler konumlanır ?

20- Türkiye' de hangi iklim kuşakları bulunmaktadır ?

21- Ege Bölgesi'ndeki nehirlerin uzunluklarını gösterir misin?

22- Ege Bölgesi'nde iklim koşulları nasıldır ?

23- Söylesene Bursa ili hangi coğrafi bölgededir ?

24- Ege Bölgesi'ndeki şehirlerin iklimlerini gösterir misin ?

25- İzmir'in komşu illerini sıralar mısın ?

26- Ankara ilinin rakımı kaçtır?

27- Manisa ilinin yüzölçümü kaçtır ?

28- Hazar Gölü hangi ilimizdedir ?

29- Hangi dağlar Akdeniz Bölgesi'ndeki şehirlerde konumlanır ?

30- Karadağ dağı hangi ilimizdedir ?

31- Türkiye'nin nüfusu kaçtır ?

32- Ege Bölgesi'nde görülen iklimler nelerdir ?

33- Konya ili hangi ülkededir?

34- Türkiye'nin komşuları hangi ülkelerdir ?

35- Ankara şehrine komşu olan şehirlerin rakımlarını gösterir misin ?

36- Harran ovası hangi coğrafi bölgemizde bulunmaktadır ?

37- Harran Ovası hangi ilimizdedir ?

38- Türkiye'nin nüfusu kaçtır ?

39- Akdeniz Bölgesi'nin toplam nüfusu ne kadardır?

40- Manisa'nın komşu illeri hangileridir ?

41- İstanbul Boğazı'nın uzunluğu kaç kilometredir?

42- Akdeniz bölgesinde bulunan dağların yüksekliklerini öğrenebilir miyim?

43- Konya şehrine komşu olan şehirlerin nüfuslarını görebilir miyim?

44- Adıyaman yüzölçümü en büyük kaçıncı ilimizdir ?

45- Toplam ilçe sayısı en fazla olan il hangisidir ?

46- Hangi nehirler Ege Bölgesi'nden geçer ?

47- Bozcaada nüfus olarak Türkiye'deki en büyük kaçıncı adadır ?

48- Yıllık ortalama sıcaklığı 15 derecenin üzerinde olan iller hangileridir ?

49- Çevresinde en fazla şehir bulunan il hangisidir ?

50- Ege Bölgesi'nden geçen nehirler hangileridir?

51- Hangi akarsular aynı anda birden fazla ilde bulunur ?

52- Hangi dağlar Akdeniz Bölgesi’nde konumlanır?

53- Hangi nehirler sadece tek bir ilde bulunur ?

54- Türkiye'nin koordinatları nelerdir ?

55- Tuzluluk oranı en yüksek olan deniz hangi coğrafi bölgededir?

56- Söyle bakalım Ege Bölgesi'nde görülen iklim nedir ?

57- Hangi göller Güneydoğu bölgesindedir?

58- Ege Bölgesi'deki illerin ilçeleri hangileridir ?

59- Ege Bölgesi'ndeki şehirlerin ilçelerinin nüfuslarını gösterir misin ?

60- Yamanlar dağı hangi şehrimizdedir ?

61- Maden ilçesinin nüfusu kaçtır?

62- Sivrice ilçesindeki dağların yükseklikleri kaçtır?

63- Ödemiş ilçesinin yüzölçümünü gösterir misin?

64- Büyükada hangi coğrafi bölgededir?

65- Heybeliada hangi ile bağlıdır ?

66- Hangi dağlar sadece tek bir ilde bulunur ?

67- Sivrice ilçesinde bulunan gölleri sıralar mısın?

68- Büyükada’da bulunan dağlar hangileridir ?

69- Ödemiş’te bulunan dağların yüksekliklerini sıralar mısın?

70- Süphan dağı hangi coğrafi bölgemizdedir ?

71- Hangi ülkeler Türkiye’ye komşudur?

72- Marmara Bölgesi’ndeki iklim koşulları nelerdir?

73- Ege Bölgesi'ndeki iklim kuşağı nedir ?

74- Ege Bölgesi’nden hangi nehirler geçer ?

75- Karadeniz’in tuzluluk oranı kaçtır?

76- Türkiye’ nin akarsu sayısı en fazla olan coğrafi bölgesi hangisidir ?

77- Türkiye’ nin nüfus yoğunluğu en az dördüncü ili hangisidir ?

78- Ege Bölgesi'nde hangi iklim yaşanır ?

79- Akdeniz Bölgesi'ndeki şehirlerin iklimleri nasıldır ?

80- Türkiye'nin çevresinde konumlanan ülkeler hangileridir ?

81- Türkiye'nin komşularının isimleri nelerdir ?

82- Ege Bölgesi'ndeki ovaların yüzölçümlerini gösterir misin ?

83- Akdeniz Bölgesi'ndeki göllerin derinliklerini öğrenebilir miyim ?

84- Hangi nehirler Karadeniz Bölgesi'nden geçer?

85- Hangi ovalar Güneydoğu Anadolu Bölgesi'nde konumlanır ?

86- Hangi göller Akdeniz Bölgesi'nde konumlanır ?

87- İzmir'in nüfusu Balıkesir'in nüfusundan ne kadar fazladır?

88- Türkiye'nin en derin gölü hangi coğrafi bölgemizde yer alır?

89- Ege Bölgesi'nde hangi ilçeler bulunur ?

90- Ege Bölgesi'deki illerin ilçelerinin nüfuslarını gösterir misin ?

91- Manisa şehrinin çevresindeki şehirlerin nüfuslarını görebilir miyim ?

92- Ankara'nın komşu şehirlerinde bulunan ovaları gösterir misin ?

93- Hangi nehirler Manisa'ya komşu olan şehirlerden geçer ?

94- Komşuları Ege Bölgesi'nde bulunan şehirlerin nüfuslarını gösterir misin ?

95- Türkiye'de karasal iklimin yaşandığı en kalabalık şehir hangisidir ?

96- Türkiye'deki denizlerin tuzluluk oranlarını gösterir misin ?

97- Türkiye'deki nehirlerin uzunluklarını görebilir miyim ?

98- İzmir'in ait olduğu coğrafi bölgede hangi iklim yaşanır ?

99- Türkiye'deki şehirlerin ortalama yağış miktarlarını gösterir misin ?

100- Türkiye'nin başkenti hangi şehirdir ?

APPENDIX 2 – Implemented Classes and Methods

Some parts from Pipeline, QuantitativeAnswer classes and generateSparqlComponents() and generateSparql () methods implemented in Java are represented.

```
public class Pipeline {

	static String sentence =" Türkiye'nin en yüksek dağı neresidir ?";
	public static void main(String[] args) throws IOException, Exception
{

	  Pipeline pipeline = new Pipeline();
	  List <String> morphAnalyzerResult =
pipeline.createMorphAnalyzerOutput(sentence);
	     String disambiguatorInput =
pipeline.createDisambiguatorInput(morphAnalyzerResult,sentence);
		String disambiguatorOutput =
pipeline.createDisambiguatorOutput(disambiguatorInput);
		System.out.println("DISAMBIGUATOR RESULT: \n");
		System.out.println(disambiguatorOutput+"\n");
		String nerInput =
pipeline.createNerInput(disambiguatorOutput);
		String nerOutput = pipeline.createNerOutput(nerInput);
		System.out.println("NER RESULT: \n");
		System.out.println(nerOutput);
		String dependencyInput =
pipeline.convertToConll(disambiguatorOutput);
		String dependencyOutput =
pipeline.createDependencyOutput(dependencyInput);
		System.out.println("DEPENDENCY RESULT:\n"+dependencyOutput);
		boolean numericInvolved = false;
		QuantitativeAnswer qa = new QuantitativeAnswer ();
		numericInvolved =
qa.checkQuantitative(nerOutput,dependencyOutput);
		if(numericInvolved==true)
		{
			String result = qa.generateSparqlComponents(sentence);
			String finalQuery =
qa.generateSparql(result,nerOutput);
			System.out.println("FINAL QUERY:\n"+finalQuery);
		}
		else
		{
		String query = pipeline.generateSparql(morphAnalyzerResult,
disambiguatorOutput, nerOutput,dependencyOutput );
		System.out.println("FINAL QUERY:\n"+query);
		}

	}
```

```
public class QuantitativeAnswer {

	static String sentence ="En çok yağış alan bölgemiz neresidir ?";
	public static void main(String[] args) throws IOException, Exception
{
		Pipeline pipeline = new Pipeline();
		  List <String> morphAnalyzerResult =
pipeline.createMorphAnalyzerOutput(sentence);
		String disambiguatorInput =
pipeline.createDisambiguatorInput(morphAnalyzerResult,sentence);
			String disambiguatorOutput =
pipeline.createDisambiguatorOutput(disambiguatorInput);
			String nerInput =
pipeline.createNerInput(disambiguatorOutput);
			String nerOutput = pipeline.createNerOutput(nerInput);
			String dependencyInput =
pipeline.convertToConll(disambiguatorOutput);
			String dependencyOutput =
pipeline.createDependencyOutput(dependencyInput);
			System.out.println("DEPENDENCY
RESULT:\n"+dependencyOutput);
			boolean numericInvolved = false;
			QuantitiveAnswer qa = new QuantitiveAnswer();
			numericInvolved =
qa.checkQuantitative(nerOutput,dependencyOutput);
			System.out.println(numericInvolved);
			String result = qa.generateSparqlComponents(sentence);
			System.out.println(result);
			String finalQuery =
qa.generateSparql(result,nerOutput);
			System.out.println(finalQuery);
	}
	public boolean checkQuantitative( String nerResult, String
depOutput)
	{
		boolean quantAns=false;
		String[] lines = depOutput.split("\n");

		for(String line : lines)
		{
			String[] tokens = line.split("\t");

			if(tokens[1].equalsIgnoreCase("kaç") ||
tokens[1].equalsIgnoreCase("kadar") || tokens[1].equalsIgnoreCase("en"))
			{
				if(tokens[3].equalsIgnoreCase("Adj") ||
tokens[3].equalsIgnoreCase("Adverb") )
				{
					int indexConnected =
Integer.parseInt(tokens[7]);
					int currentIndex =
Integer.parseInt(tokens[0]);
					if(lines[indexConnected-
1].split("\t")[3].equalsIgnoreCase("Noun") ||
lines[currentIndex+1].split("\t")[3].equalsIgnoreCase("Noun")
                                                ||
lines[currentIndex].split("\t")[3].equalsIgnoreCase("Adj"))
					{
						quantAns = true;
```

```
                                }

                            }
                        }

                }

                return quantAns;
        }

        public String generateSparqlComponents(String sentence) throws
IOException, Exception
        {

                StringBuilder resultList = new StringBuilder();
        String testFilePath =
"C:/Users/Ceren/workspace/Geo_Interface/src/trainingData/sentenceTestClass
.arff";
                FileInputStream fs = new FileInputStream(testFilePath);

                String content = IOUtils.toString(fs,
Charset.defaultCharset());
                String [] parts = content.split("@data");
                FileWriter fwriter2 = new FileWriter(testFilePath,false);
                fwriter2.write(parts[0]+""+"@data\n");
                fwriter2.close();

                Classifier classifier = new MultilayerPerceptron();
            Instances train = new Instances(new BufferedReader(new
FileReader("C:/Users/Ceren/workspace/Geo_Interface/src/trainingData/senten
ceTrainClass.arff")));


            FileWriter fwriter = new
FileWriter("C:/Users/Ceren/workspace/Geo_Interface/src/trainingData/senten
ceTestClass.arff",true); //true will append the new instance
            fwriter.write('"'+sentence+'"'+",?,?,?,?,?");//appends the string
to the file
            fwriter.close();

            Instances test = new Instances(new BufferedReader(new
FileReader(testFilePath)));
            train.setClassIndex(1);
            test.setClassIndex(1);
            int lastIndex = train.numAttributes() - 1;
            for (int i =1; i <= lastIndex ; i++ )
            {

            train.setClassIndex(i);
            test.setClassIndex(i);

             StringToWordVector filter = new StringToWordVector();
              filter.setInputFormat(train);
              filter.setLowerCaseTokens(true);
              FilteredClassifier fc = new FilteredClassifier();
              fc.setFilter(filter);
              //specify base classifier
              fc.setClassifier(classifier);
              //Build the meta-classifier
```

```
          fc.buildClassifier(train);
         double index = fc.classifyInstance(test.instance(0));
         test.instance(0).setClassValue(index);


       resultList.append(test.instance(0).stringValue(i)+",");
         }

     return resultList.toString();

       }

       public String generateSparql(String result, String nerOutput) throws
OWLOntologyCreationException
       {

              StringBuilder finalQuery = new StringBuilder();
              Pipeline2 p2 = new Pipeline2();
              String ontologyName = p2.returnOntologyName();
              String[] components = result.split(",");
               String target_class = components[0];
               String entity_class = components [1];
               String data_prop = components[2];
               String obj_prop = components[3];
               String agg_func = components[4];

               String[] nerLines = nerOutput.split("\n");   //NER OUTPUTA
BAKTI?I YER
                    String ner_word = null;
                    String[] list_ner_words=null;
                    for(String nerLine : nerLines)
                    {
                           if(nerLine.contains("B-LOCATION"))
                           {

                                  String[] ner_words = nerLine.split(" ");

                                  list_ner_words =ner_words[1].split("\\+");
                                  ner_word = list_ner_words[0];

                           }

                    }



              if(agg_func.equalsIgnoreCase("count"))
              {
                     if(!target_class.equalsIgnoreCase("null"))
                     {
                     finalQuery.append("SELECT
("+agg_func.toUpperCase()+"(?x) as ?total)\nWHERE {");
                     finalQuery.append(" ?x"+" rdf:type
"+ontologyName+":"+target_class+" .\n");

                     if(!entity_class.equalsIgnoreCase("null"))
                     {
                            finalQuery.append("?y"+" rdf:type
"+ontologyName+":"+entity_class+" .\n");
```

```
                        }
                        if(!obj_prop.equalsIgnoreCase("null"))
                        {
                                finalQuery.append("?x
ins:"+obj_prop+" ?y .\n");
                        }
                        if(ner_word!=null)
                        {
                      ner_word = Normalizer.normalize(ner_word,
Normalizer.Form.NFD).replaceAll("\\p{Mn}", "");
                        finalQuery.append("FILTER(regex(str(?y),");
                        finalQuery.append("\""+ner_word+"\",");
                        finalQuery.append("\""+"i"+"\")) }");
                        }
                        }
                        else if((!entity_class.equalsIgnoreCase("null")) &&
(!data_prop.equalsIgnoreCase("null")))
                        {
                                finalQuery.append("SELECT
("+agg_func.toUpperCase()+"(?var) as ?total)\nWHERE {");
                                finalQuery.append(" ?x"+" rdf:type
"+ontologyName+":"+entity_class+" .\n");
                                finalQuery.append("?x
ins:"+data_prop+" ?var.\n");
                                ner_word = Normalizer.normalize(ner_word,
Normalizer.Form.NFD).replaceAll("\\p{Mn}", "");
                                finalQuery.append("FILTER(regex(str(?x),");
                                finalQuery.append("\""+ner_word+"\",");
                                finalQuery.append("\""+"i"+"\")) }");

                        }

                }

                else if(agg_func.equalsIgnoreCase("max")||
agg_func.equalsIgnoreCase("min"))
                {
                        agg_func =
Normalizer.normalize(agg_func.toUpperCase(),
Normalizer.Form.NFD).replaceAll("\\p{Mn}", "");

                        if(!target_class.equalsIgnoreCase("null"))
                        {
                                finalQuery.append("SELECT ?x ?var \nWHERE {");
                                finalQuery.append("?x"+" rdf:type
"+ontologyName+":"+target_class+" .\n");
                                if(!data_prop.equalsIgnoreCase("null"))
                                {
                                        finalQuery.append("?x
ins:"+data_prop+" ?var .\n{");
                                        finalQuery.append("SELECT
("+agg_func.toUpperCase()+"(?val) as ?var)\nWHERE {");
                                        finalQuery.append(" ?x"+" rdf:type
"+ontologyName+":"+target_class+" .\n");

if(!entity_class.equalsIgnoreCase("null"))
                                        {
                                                finalQuery.append("?y"+" rdf:type
"+ontologyName+":"+entity_class+" .\n");
```

```
                                    }
                                    if(!obj_prop.equalsIgnoreCase("null"))
                                    {
                                          finalQuery.append("?x
ins:"+obj_prop+" ?y .\n");
                                    }
                                    finalQuery.append("?x
ins:"+data_prop+" ?val .\n");
                                    if(ner_word!=null)
                                    {
                                  ner_word = Normalizer.normalize(ner_word,
Normalizer.Form.NFD).replaceAll("\\p{Mn}", "");

finalQuery.append("FILTER(regex(str(?y),");
                                    finalQuery.append("\""+ner_word+"\",");
                                    finalQuery.append("\""+"i"+"\")) }}}");
                                    }
                                    else
                                    {
                                          finalQuery.append("}}}");
                                    }

                              }
                        }
                  }


            return finalQuery.toString();
      }

}
```