Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Håkon Kristian Lem Mardal

Mímir: a Norwegian Question Answering System for Searching Wikidata

Master's thesis in Informatics
Supervisor: Trond Aalberg
June 2022

NTNU
Norwegian University of
Science and Technology

Håkon Kristian Lem Mardal

Mímir: a Norwegian Question Answering System for Searching Wikidata

Master's thesis in Informatics
Supervisor: Trond Aalberg
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Mímir: a Norwegian Question Answering System for Searching Wikidata

Håkon Kristian Lem Mardal

June 13, 2022

Abstract

Semantic Web Knowledge Bases contains massive amounts of information which can be easily accessed by creating natural language interfaces. Question Answering systems are an extension to natural language interfaces with the goal of answering questions a user can conceive. A problem with such semantic web interfaces is that they usually require that the user knows the English language. However, some Knowledge Bases include multi-linguistic support such as Wikidata which we can take advantage of. The over-arching goal of this thesis is therefore to leverage the Norwegian data available in Wikidata to create a simple Norwegian specific question answering system over Wikidata and illuminate the challenges of implementing a Norwegian specific system.

The accuracy tests and system analysis performed shows the validity of the presented approach. The inverted indexes provided also open up the path for further development using them and lay the footwork towards Norwegian specific solutions for Question answering systems over Knowledge Bases.

Sammendrag

Semantiske web kunnskapsbaser inneholder veldig store datamengeder av information som kan lett bli aksessert gjennom naturlig språk grensesnitt. Systemer for å svare på spørsmål er en utvidelse til naturlig språk grensesnitt med målet å svare på alle spørsmål en kan ha. Et problem med slike semantisk web grensesnitt er at de vanligvis bare er tilgjengelige på engelsk. Det finnes noen kunnskapsbaser i dag som støtter flere språk sånn som Wikidata som vi kan ta fordel av. Hovedmålet for denne masteroppgaven er derfor å bruke de norske dataene tilgjengelig i Wikidata til å lage et enkelt norsk spesifikt system for å svare på brukerspørsmål og belyse utfordringer ved å implementere et slikt norskt system.

Nøyaktighetstestene og systemanalysen utført viser validiteten til den presenterte metoden. De norske inverterte indeksene kan bli brukt for videre utvikling av norske systemer og viser fram veien mot norsk-spesifikke løsninger for systemer for spørsmåls-svaring over kunnskapsbaser.

Preface

This Master Thesis was submitted to the Norwegian University of Science and Technology (NTNU), Department of Computer Science(IDI) as part of the course IT3920 - Informatics Postgraduate Thesis: Database Management and Search.

I would like to thank my advisor Trond Aalberg for his guidance, ideas and feedback during the course. I also want to thank my friends and family for their relentless support throughout the whole period of my studies.

Contents

Abstract . . . . . **iii**
Sammendrag . . . . . **v**
Preface . . . . . **vii**
Contents . . . . . **ix**
Figures . . . . . **xi**
Tables . . . . . **xiii**
1 Introduction **1**
1.1 Purpose 1
1.2 Research Goal 3
1.3 Contributions 3
1.4 Thesis Structure 4
2 Background and Related Work **5**
2.1 Background Theory 5
2.1.1 Semantic Web 5
2.1.2 Querying SPARQL VS Querying SQL 9
2.1.3 Natural Language Interfaces and the Question Answering Task 9
2.1.4 Knowledge Base Question Answering 10
2.1.5 Four Natural Language Processing Approaches 12
2.1.6 Available Norwegian NLP Tools and Resources 16
2.2 Review of Related Work 17
2.2.1 SINA 17
2.2.2 Athena 19
2.2.3 FREyA 21
2.2.4 Querix 23
2.2.5 Aqqu 24
3 Method **27**
3.1 Research Methodology 27
3.2 Approach 28
3.2.1 Pre-processing the Question 29
3.2.2 Entity Matching 30
3.2.3 Triple Candidate Generation 31
3.2.4 Relation Matching 32
3.2.5 Ranking 33

3.2.6 Template Mapping and Answer SPARQL Execution 33
3.3 Evaluation Methods 34
4 Implementation 35
4.1 Python and Python Libraries 35
4.2 Wikidata SPARQL Endpoint 36
4.3 Qlever Query Engine and SPARQL Endpoint 36
4.3.1 Official Written Languages Bokmål and Nynorsk 36
4.3.2 Norwegian Alias Entity Index 37
4.3.3 Norwegian Alias Relation Index 37
4.3.4 Answer format 38
4.4 Simple Norwegian Questions for Wikidata 38
4.5 Architecture Overview 39
5 Evaluation and Discussion 41
5.1 Datasets 41
5.2 Evaluation 42
5.2.1 Results 42
5.3 Analysis 44
5.3.1 Error Analysis 44
5.4 Discussion 46
5.4.1 RDF Graph Data Structure and Norwegian 46
5.4.2 Norwegian Methods 46
5.4.3 Available Norwegian NLP Tools 46
5.4.4 Limitations of KB Datasets 46
5.4.5 Limitations of Testsets 47
5.4.6 Findings 48
6 Conclusion 49
6.1 Conclusion 49
6.2 Further Work 50
6.2.1 User Feedback 50
6.2.2 Complex Norwegian Question Answering 50
6.2.3 Norwegian Testsets 50
6.2.4 Norwegian Knowledge Base Expansion 50
6.2.5 Norwegian Entity recognition and Linking tools 50
Bibliography 51

Figures

2.1 Wikidata Entity page for the United States of America including labels and aliases for English and Norwegian. 7
2.2 Wikidata Predicate/property page for the nationality relation including labels and aliases for English and Norwegian. 8
2.3 Example simple question answering over Wikidata in Norwegian . . 10
2.4 Example complex question answering over Wikidata in Norwegian 11
2.5 Overview over the SODA Approach. Source: [3] 12
2.6 A visualization of the CoreNLP dependency parse of a sentence . . . 14
2.7 Financial Domain Ontology applied in Athena. Source: [15] 15
2.8 Example grammar rules present in TR Discover 15
2.9 Overview of the SINA engine 18
2.10 Overview of the Athena engine 20
2.11 Example use of the Athena Ontology Evidence Annotator 20
2.12 Freya: Validation process of potential ontology concepts through user interaction. Source:[20] 22
2.13 Aqqu: query templates and example candidates with corresponding questions 25

3.1 Overview of the Mímir Pipeline 28
3.2 Simple question answering over wikidata in Norwegian using Mímir approach 34

4.1 Overview of the Mímir Implementation 39

Tables

3.1 Table showing different N-grams of a Norwegian question sentence which are used to link entity mentions to entities in the KB 29
3.2 Table showing example linked entity aliases to N-grams entity mentions in the question. 31
3.3 Table showing example linked relation aliases to lemmatized relation mentions in the question. 33

5.1 Table showing accuracy results for the two evaluation testsets used in this project. 42
5.2 A split table showing 5 example results from the Norwegian translated SimpleQuestions for Wikidata. 43
5.3 A split table showing 5 example results from the Mímir analysis testset. 43

Chapter 1

Introduction

This chapter provides an introduction to the purpose of this research project, the planned contribution of this work and an overview to the structure of the thesis.

1.1 Purpose

When I was growing up in the 90s and 2000s, the Internet was a useful tool when searching for answers to questions one might have. As I grew older, structured information resources such as the Wikipedia (a online encyclopedia) became frequently utilized when querying information for school related projects. Natural language interfaces provided by Google or Wikipedia to search such resources are easy to use because they only require the user to have some notion of the existing semantic information contained in the database. If Wikipedia did not provide a natural language interface, one would have to learn complex query languages like SQL or SPARQL, and learn the schema of the Wikipedia database to be able to retrieve such information.

With the introduction of the Semantic Web which is a vast, machine readable and highly interconnected web of data, there is a growing need for natural language interfaces for databases (NLIDB) that can answer queries of semantic web data/Linked Data[1]. These interfaces also serve as means to give common users access to the Semantic Web, in a similar way as it has existed for the document-based web for the duration of the Internet's history.

A prerequisite for using such interfaces is to learn the English language as large amount of the interfaces to the semantic web are only made available in English. Most of the advanced language technology - even today is only available in English or in other languages with sizable amount of users. However there are efforts made today towards supporting more languages. The latest Question Answering over Linked Data challenge (QALD-9) [1], which is a state of the art benchmark for evaluating open-domain Question Answering systems over Linked

[1]Linked data is the data storing structure used in the Semantic Web

Data, includes multilingual support for 11 different languages such as English, Spanish, German, Italian, French, Dutch, Romanian or Farsi.

Norwegian is an example of a language with relatively few users with a population of 5,4 million. [2] The Norwegian Language Council [2] advised that Norwegian versions of advanced language technology products should be made available in Norwegian to protect and ensure the usage of Norwegian in the future. The problem is that there exists few Natural Language Processing resources and tools with Norwegian support. However there exist research interest groups such as NLP Norway [3] that keep track of available NLP tools and resources with Norwegian support at their Github Page[4].

Even though Semantic Web search engines and Question Answering systems of Semantic Web databases has existed for a long time now, and this field has enjoyed large quantities of present research such as [3–8], there still does not exist - to the best of my knowledge a Norwegian specific Question Answering system for Semantic Web data. This master thesis is an attempt to illuminate general approaches used in creating natural language interfaces and Question Answering systems, present a Norwegian Question Answering System and display the challenges implementing such a system.

[2]Population Statistics reported by Statistisk sentral byrå, updated 19th of May, 2022. Link: https://www.ssb.no/en/befolkning/folketall/statistikk/befolkning

[3]NLP Norway Facebook Link: https://www.facebook.com/groups/nlpnorway

[4]NLP Norway Github Page: https://github.com/web64/norwegian-nlp-resources.

1.2 Research Goal

Goal *Investigate how to implement a Norwegian Question Answering system for a Knowledge Base with semantic data that translates Norwegian natural language queries to SPARQL, querying the knowledge base for correct answers.*

Hypothesis *Question Answering systems over knowledge bases that store semantic data with multilingual support can be adapted to answer simple natural language questions in Norwegian by leveraging the topology of Linked Data, and the Norwegian entity- and relation aliases present in the Knowledge Base to match natural language to entities and relations.*

Research Question 1 *How can the data present in the Semantic Web Knowledge Base be used to answer natural language queries in Norwegian?*

Research Question 2 *What challenges are present to answer user queries in Norwegian over a Knowledge Base?*

Research Question 3 *How can we evaluate the performance of our Norwegian Question Answering system?*

1.3 Contributions

The project provides a Norwegian Question Answering system over Wikidata, and presents how it can be used for answering simple natural language queries.

1. *Provides a novel question answering system for translating simple Norwegian natural language queries to SPARQL queries based on existing approaches in English*
2. *Provides an entity index dataset for Norwegian linking Norwegian aliases to entities present in Wikidata*
3. *Provides a relation index dataset for Norwegian linking Norwegian aliases to properties present in Wikidata*
4. *Provides a Norwegian evaluation set of simple questions and wikidata answer pairs, manually crawled and inspired by SimpleQuestions For Wikidata dataset.*

1.4 Thesis Structure

Chapter 2 Chapter 2 consist of background knowledge explaining basic concepts around semantic web and NLP task of converting NL to database query language as well as related work in the domain.

Chapter 3 Chapter 3 lays out the methodology and approach.

Chapter 4 Chapter 4 presents the implementation of the architecture and the different technologies needed for building such a system.

Chapter 5 Chapter 5 lays out the evaluation settings and the different testing experiments for the project as well as discusses the the merits of the work, summarizes the findings and the limitations of the project.

Chapter 6 Chapter 6 concludes the work done and presents future work for further improvements.

Chapter 2

Background and Related Work

This chapter contains the theoretical material needed to understand the project and the different building blocks which are necessary to cover the theoretical basis. The chapter starts with an overview over Semantic Web technologies, the general task of creating natural language interfaces for databases (NLIDB), continuing to my specific Knowledge Base Question Answering task, natural language processing approaches and reviews related work present in the literature.

2.1 Background Theory

Mímir is accounted as the wisest god in Norse mythology and a trusted advisor to Odin. Mímir guarded the "Well of Knowledge", often named the "Well of Mímir" from which he would drink wisdom from. My Question Answering system is named Mímir to personify the system as a gateway between the users and the vast information present in Semantic Web Knowledge Bases.

2.1.1 Semantic Web

The Semantic Web provides a semantic way of representing, storing and querying information. Tim Bernes-Lee the creator of the World Wide Web, defined the Semantic Web as *"a web of data that can be processed directly and indirectly by machines"*. The main idea of the Semantic Web is to support a distributed web at the level of the data (entities, concepts) rather than at the level of the presentation (documents). What this means is instead of storing web pages aka documents as singular values on the web, the Semantic Web stores the underlying concepts, entities and data as individual resources confined in the documents, making them building blocks of semantic information. This way, links (hyperlinks) and relations can exist between resources that explain how they are related semantically, and the same resources can be linked and matched across the web.

The World Wide Web functions as a provider of documents (websites) using hyperlinks (URLs), the Semantic Web provides resources using hyperlinks. These resources are semantic in nature meaning they describe some entity which can

be physical or abstract concepts and are connected and point to one another by using global references called Uniform Resource Identifiers (URI) . These entities and concepts can be grouped, linked and expanded across heterogeneous websites to get a consistent web of their meanings and relations. Such entities can be for instance persons, organizations, objects, addresses, numbers. This way we can retrieve and present consistent information on the internet based on its semantic meaning.

To be able to represent and store machine-readable relations in the distributed web of data, the data model Resource Description Framework (RDF) is used as a format to create RDF triple-stores that describe how entities relate and together constitute linked datasets. A RDF triple consists of a *Subject*, *Predicate* and *Object*[1] that models machine-readable data relations. RDF triple can also be described in graph terms as two nodes and an edge between them. An example triple describing marriage;

$< S > Bill\_Gates, < P > married\_to, < O > Miranda\_Gates.$

An example triple statement using literals, aka property values like strings, numbers, boolean, etc. In this case string type:

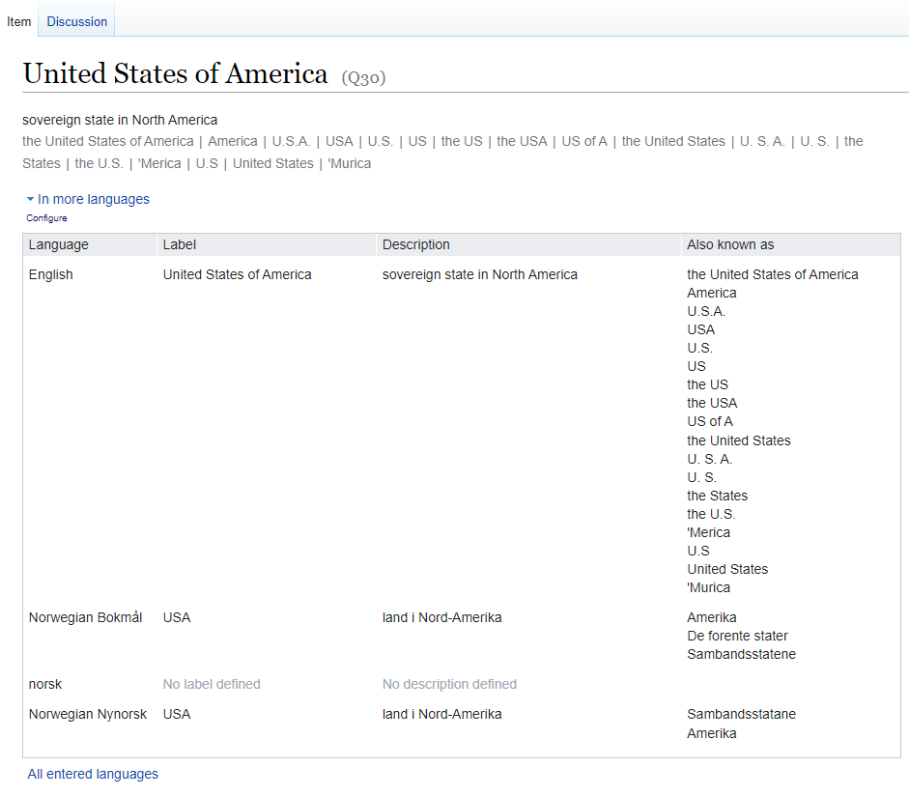
$< S > Bill\_Gates, < P > nationality, < O > America.$

To be able to define hierarchies to describe how these URIs are connected in a certain subject or domain, ontologies are defined using Resource Description Framework Schema (RDFS) and Web Ontology Language (OWL) which are used as relation vocabularies in the Semantic Web technology stack. An ontology is a machine-readable representation of a domain with concepts, entities and relations between them. The ontologies provides a way to recognize terms and its surrounding domain environment in a natural language query which can then be used to correctly map the input query to a database query language.

RDF literals can also be tagged with a language tag, for instance the literal from the last example regarding Bill Gates, where "America@en" can be tagged with corresponding English language tag, "@en" at the end of the string. A Norwegian tag for the literal "America" can be "Amerika@nb" where "@nb" are initials for the Norwegian standard written language "Norsk Bokmål". These language tags can be used to filter database queries to return language specific literals.

DBpedia [9] and Wikidata [10] aim to create multilingual knowledge base based on RDF-stores encompassing all general knowledge so that humans can edit and machines can read on the Semantic Web. These knowledge bases contains structured information from the general online encyclopedia Wikipedia. The term "knowledge base" is often used as a synonym to the term "knowledge graph" which describes a real world perception of entities and their interrelations organized in a graph. These RDF-stores are multilingual in the sense that they store RDF information for many languages. Entities aka RDF subjects or objects have many linked triple statements linking them to information in many different languages. Note that this also includes multiple synonyms/aliases in the respective languages.

[1] Objects can also be literals in a RDF triple



Item Discussion

United States of America (Q30)

sovereign state in North America

the United States of America | America | U.S.A. | USA | U.S. | US | the US | the USA | US of A | the United States | U. S. A. | U. S. | the States | the U.S. | 'Merica | U.S | United States | 'Murica

In more languages

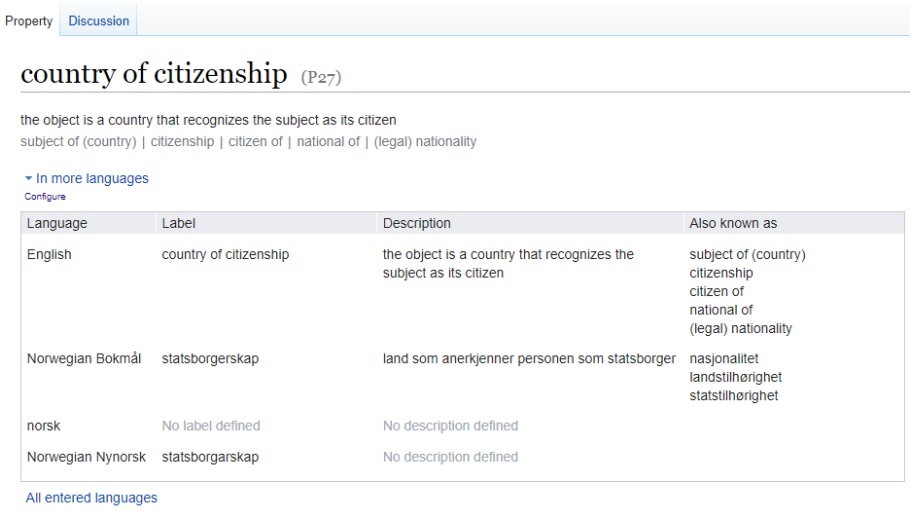
Configure

Language	Label	Description	Also known as
English	United States of America	sovereign state in North America	the United States of America America U.S.A. USA U.S. US the US the USA US of A the United States U. S. A. U. S. the States the U.S. 'Merica U.S United States 'Murica
Norwegian Bokmål	USA	land i Nord-Amerika	Amerika De forente stater Sambandsstatene
norsk	No label defined	No description defined	
Norwegian Nynorsk	USA	land i Nord-Amerika	Sambandsstatane Amerika

All entered languages

Figure 2.1: Wikidata Entity page for the United States of America including labels and aliases for English and Norwegian. Note that the table has not been expanded to show other supported languages as well. RDF triple statements about the entity can be found further down on the Wikidata webpage, which are not shown in this figure.
Available at: *https://www.wikidata.org/wiki/Q30*

Wikidata relations aka RDF predicates are also stated using labels and aliases for many different languages. A Wikidata Entity has many statements connected to it, which are essentially connected RDF predicates and other Wikidata entities or literals. Note that Norwegian has two official languages, Norwegian Bokmål and Norwegian Nynorsk.



Property Discussion

country of citizenship (P27)

the object is a country that recognizes the subject as its citizen

subject of (country) | citizenship | citizen of | national of | (legal) nationality

▾ In more languages

Configure

Language	Label	Description	Also known as
English	country of citizenship	the object is a country that recognizes the subject as its citizen	subject of (country) citizenship citizen of national of (legal) nationality
Norwegian Bokmål	statsborgerskap	land som anerkjenner personen som statsborger	nasjonalitet landstilhørighet statstilhørighet
norsk	No label defined	No description defined	
Norwegian Nynorsk	statsborgarskap	No description defined	

All entered languages

Figure 2.2: Wikidata Predicate/property page for the nationality relation including labels and aliases for English and Norwegian. Note that the table has not been expanded to show other supported languages as well.
Available at: *https://www.wikidata.org/wiki/Property:P27*

2.1.2 Querying SPARQL VS Querying SQL

RDF query languages are based on querying graph patterns in the form of RDF triples accessing a web of linked data. The triples consist of subject, predicate and object resources. To query using a triple, you can replace a resource in any of the positions with a variable indicated by the question mark character **?**. If you want to know who directed the movie "The Dark Knight", the graph pattern can look like this:

```
?director dir:directedMovie movie:TheDarkKnight.
```

Of course you can have multiple RDF triples to query a larger graph pattern. A SPARQL query can look like this (ignoring prefixes) to find director and actor where the actor name is Christian Bale:

```
SELECT ?director ?actor
WHERE {
?director dir:directedMovie ?movie.
?actor actor:actedInMovie ?movie.
?actor actor:name "Christian Bale".
}
```

The general idea is that you can query sub-graphs using a set of RDF statements (graph patterns) for the information you need.

SQL relies on the relational algebra of joins and primary key - foreign key references between tables to answer queries. Example SQL query for querying similar information as the SPARQL example defined previously:

```
SELECT Director.name, Actor.name
FROM Director
JOIN Movie
  ON Director.ID = Movie.director_id
JOIN Actor
  ON Movie.actor_id = Actor_id;
WHERE Actor.name = "Christian Bale"
```

So even though SPARQL and SQL share similar syntax such as SELECT, WHERE, GROUP BY and ORDER BY, the underlying query structure is very different.

2.1.3 Natural Language Interfaces and the Question Answering Task

Natural language interface for databases (NLIDB) are interfaces that take some natural language query, often keywords or sentences as input and automatically produces results from a database by employing database query language like SQL or SPARQL. The advantage of implementing such interfaces is that it eliminates the need to learn complex querying languages and database schemas for the users. The problem is of course to create decent implementations that can produce adequate results from ambiguous natural language queries. Popular search engines on the internet like Google, Bing and Duck Duck Go! all provide natural language

interfaces that allow queries through text fields. These search engines provides mostly relevant documents, but also possess support for Knowledge Bases which can supply individual data items as answers to query questions like the ones that we are interested in this thesis.

The question answering task consists of automatically answering questions and statements posed by users in a natural language format stating their information need. By using natural language processing tools and methods to translate natural language queries into machine readable database query languages, we can retrieve answers from databases for users that does not know query languages nor know how to query databases. The question answering task is used to test and evaluate existing search engines by providing datasets of questions and expected answers/database queries. These datasets are often structured with an increasing level of difficulty.

2.1.4 Knowledge Base Question Answering

Question Answering over Knowledge Bases, often called Question Answering over Knowledge Graphs is an extension to the general Question Answering task involving using large Knowledge Bases such as DBPedia [9] or Wikidata [10] as information hubs which are queried for possible answers. Querying such large datasets present in these knowledge graphs poses additional challenges. Wikidata contains $>$ 90 million entities (nodes) and $>$ 1150 million statements (edges) which makes the lookup and ranking for candidate answers challenging.

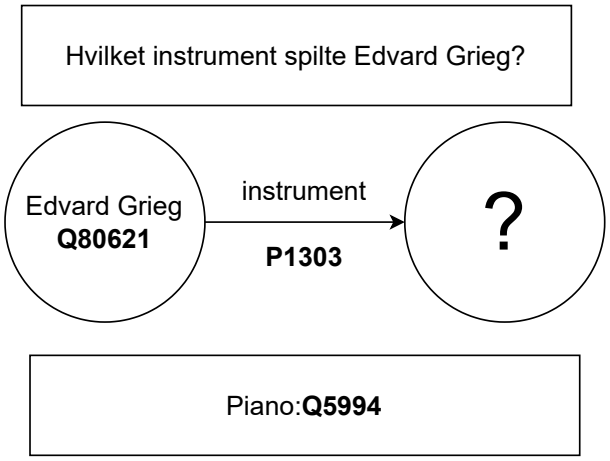


Figure 2.3: Simple question answering over Wikidata in Norwegian, resulting in a wikidata entity Piano.

Figure 2.3 shows a simple question answering over Wikidata example in Norwegian. By identifying the entity and the relation we want to query for, we can execute a SPARQL query that finds object variables where the statement Edvard Grieg -> instrument is true, resulting in the instrument entity we were looking for.

The character "Q" in combination with a unique number constitute entities identification in Wikidata, in this example Edvard Grieg has the id "Q80621". Similarly "P1303" identifies the edge statement "instrument" where "P" identifies it as a property (predicate) relation. Note that all the entries from Wikidata in this example are in Norwegian, as Wikidata is a multilingual supportive knowledge base.

Complex question answering begins when we ask questions that require more than a single RDF triple to answer the question. In such a query you can have an intermediate entity with relations that points to additional entities. An example of such a question answering task is present in figure 2.4

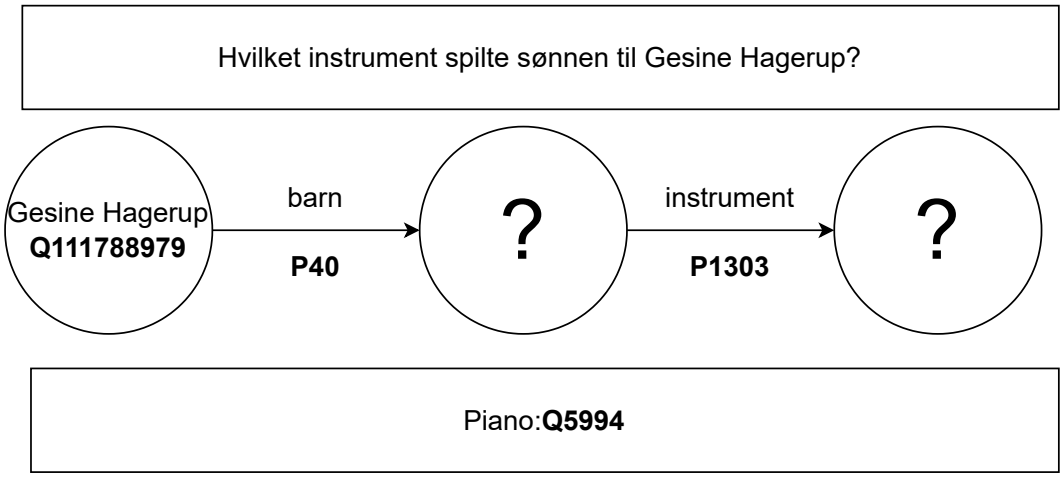


Figure 2.4: Complex question answering over Wikidata in Norwegian, requiring a sub-graph of additional entities and properties resulting in a Wikidata entity Piano.

Entity Recognition and Entity Linking

To be able to produce answers from a knowledge base, entities (and predicates) need to be recognized in the input query and linked to existing entities (and predicate relations) in the Knowledge Base. When linking the entities to the textual mentions it is important to rank and disambiguate the candidate entities to return the most correct answer. Entity indexes, aka inverted indexes of the entities can be applied to increase performance and speed-up the process of getting the correct candidates, instead of executing time consuming SPARQL queries during run-time. Inverted indexes can be built using the available aliases for entities from Wikidata which are showcased in figure 2.1. For English there exists > 98 million Wikidata entity aliases, while for Norwegian Bokmål there exists only > 7 million entity aliases. For Norwegian Nynorsk there exists <6 million entity aliases. The exact Wikidata SPARQL query for getting these numbers are shown in the implementation chapter 4.

Externally trained entity annotator and entity linking systems can exist separ-

ately for entity retrieval such as TAGME [11] and ELR (Entity Linking incorporated Retrieval) [12]. Such systems can be applied to solve the sub-problem of entity recognition and linking in question answering systems over Knowledge Bases. The problem is finding such systems that exist for Norwegian specifically.

2.1.5 Four Natural Language Processing Approaches

When looking at possible approaches in natural language for translating the actual text, they can be characterized as four main categories that are either used alone or in some combination. These four general approaches are based on keywords, parsing, grammar and machine learning.

Keyword Approach

The simplest and the most common approach is the keyword-based approach. This approach utilizes short keywords that can be mapped to entities and data values present in the database and ranked according to likely matches. An inverted index is created from the data and metadata, which is used to examine known terms in this index to find a match. Simplicity and customization are big advantages when applying this approach. Example input queries are short describing keywords for instance "highest mountain world", rather than complete sentences like "what is the highest mountain in the world called?".

SODA [3] is an example of a well-performing keyword approach that utilizes a graph pattern matching algorithm to link keywords with a metadata model of the data warehouse. The metadata model is used to map keywords to tables containing wanted information by applying domain ontologies that classify keywords to appropriate domains. The SODA pipeline is described in figure 2.5.

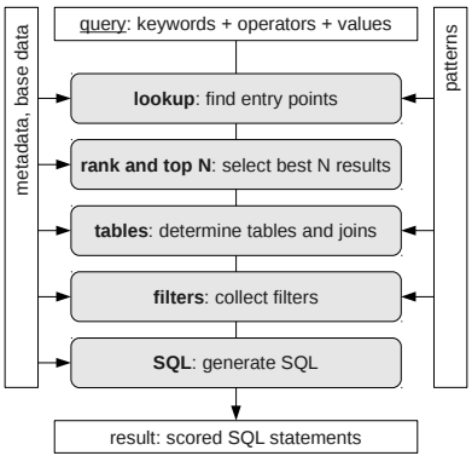


Figure 2.5: Overview over the SODA Approach. Source: [3]

Pattern-based approach is worth mentioning as it is an extension to the keyword

approach. In an additional step the keywords are analyzed for predefined patterns that are related to general categories of keywords. The found relations and patterns in the keywords are then formulated into a database query.

Parsing Approach

In this approach systems parse the input query, often using existing natural language processing tools such as NTLK [13] or Stanford CoreNLP [14]. These tools provide Part of Speech (PoS) annotations which classify words with grammatical properties in sentences. Then the tools can use the grammatically rich sentences to produce graph trees, known as parse trees with information describing individual words and grammatical relations. From analyzing the semantic meaning these relations, corresponding rules (similar to patterns-based) can be mapped from the parsed input query and used to generate valid database queries.

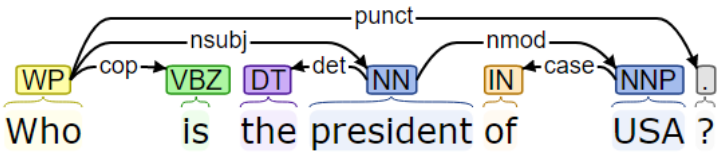


Figure 2.6: A visualization of the CoreNLP dependency parse of a sentence using their web demo available at: *https://corenlp.run*.
PoS tags: *WP*=wh-pronoun, *VBZ*=Verb, *DT*=determiner, *NN*=noun singular or mass, *IN*=preposition, *NNP*=proper noun singular.
Dependency tags: *punct*=punctuation, *nsubj*=nominal subject, *cop*=copula, *det*=determiner, *nmod*=nominal modifier, *case*=case marking.

Some systems apply this methodology by relying solely on the parse trees like NaLIR (Natural Language Interface for Relational databases) [5] which translates natural language into SQL and is able to handle aggregation functions, nesting and various types of joins.

Other systems combine parsing with some other element like Athena [15], which is ontology driven NLI for relational databases. Athena interprets NLQ with the domain specific ontology shown in figure 2.7 and relies on mappings between the ontology and the SQL database in order to work.

Grammar Approach

Although the parsing approach has some usage of grammar, the grammar approach is distinctly different from the previously mentioned approaches. This approach applies a fixed set of grammar rules that constrains the types of questions you can ask a database. Through these rules, a grammar system can guide the user into creating their queries, and in the process also expose the user to more of the database, making pretty accurate queries. The problem with this approach is that the rules are extremely domain-specific and have to be created by hand for each individual case, while other approaches can be generalized to encompass more general queries. However it is important to stress that the other approaches are to some degree also bound to this disadvantage.

An example of this approach is TR Discover [7] which creates a First Order

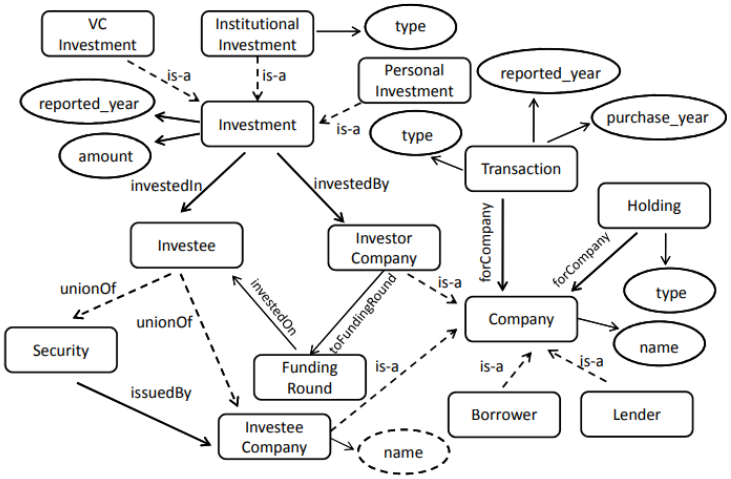


Figure 2.7: Financial Domain Ontology applied in Athena. Source: [15]

Logic representation[2] of the input query as an intermediate language. The system uses the intermediate language to propose auto-completion and predictions to the user while inputting the query. These suggestions are generated from the relationships and entities present in the database. The rules which are still valid to the input query are used to create a parsetree, which is then traversed and translated either to SQL or SPARQL.

G1: NP → N
G2: NP→ NP VP
G3: VP → V NP
Lex1: N[TYPE=drug, NUM=pl, SEM=<λx.drug(x)>] → 'drugs'
Lex2: V[TYPE=[drug,org,dev], SEM=<$\lambda X\ x.X(\lambda y$.dev_org_drug(y,x))>, TNS=past, NUM=?n] → 'developed by' /*In general, TYPE=[subject_constraint, object_constraint, predicate_name]*/
Lex3: V[TYPE=[org,country,hq], NUM=?n] → 'headquartered in'

Figure 2.8: Example grammar rules present in TR Discover. Grammar rule G3 indicates that a verb phrase (VP) contains a verb (V) and a noun phrase (NP). The Lexical entries Lex1-3 are used to map words to their database entity counterpart. Source:[7]

Machine Learning Approach

Machine Learning approaches to NLI use very rudimentary schemas. They are linguistically- or ontology-based, and can be made to work generally across the board, fitted to any training/test case one can desire. The most common ones are based on neural network models. In current research, more and more systems

[2]FOL representation is a concise way to represent natural language statements

include machine learning in their translation process as found in a comparative study of recent NLIDBs [16]. Translation from NL to database query languages like SPARQL and SQL can be formulated as a supervised machine learning problem, giving advantages such as greater variability in query expressions, letting users formulate flexible queries, compared to stiff and restricted queries present in the previously described approaches.

An example of a state of the art approach using evolutionary algorithms leveraging ontologies named EvolNLQ, is presented in the PhD dissertation by Sebastian Schrage [17] which describes a general learning framework that creates intelligent agents that learn to solve NLI to SPARQL translations.

2.1.6 Available Norwegian NLP Tools and Resources

As introduced in section 1.1 NLP Norway keeps track of available NLP tools and resources at their Github page[3]. This section is a browsing of the different tools and resources available that might be useful in our Question Answering system. SpaCy is a NLP library with support for Norwegian Bokmål with components such as parsers, lemmatizers, named entity recognition and more. Their implementation is based on the Norwegian Dependency Treebank [18] which is a syntactic treebank for Norwegian Bokmål and Nynorsk. At the github page there also exists different experimental models for different uses such as named entity recognition, Bert models, Norwegian Word vectors, sentiment analysis and more. As for corpuses they include a Norwegian stop-word dataset and general purpose Norwegian web corpus, but no datasets useful for Knowledge Base Question Answering - like an index for linking mentions to KB items. Even though Named Entity recognition tools exists for Norwegian, these are trained on news corpuses and are generally not compatible for a general Knowledge Base encompassing vast amount of knowledge. The entity recognition tools do not solve the problem of actually linking the mentions to the KB items.

[3]NLP Norway Github page: https://github.com/web64/norwegian-nlp-resources

2.2 Review of Related Work

In this section comes a review of recent search engines for graph databases as well as relational databases, and Question Answering systems that employ the different approaches previously explained. Specifically, what other research has been conducted in the area and how they motivate my work. All the presented systems evaluate on some performance measures like recall, accuracy or precision. This review was vital for me to decide which approaches was worth pursuing in this project given that we want to create a Norwegian system.

2.2.1 SINA

SINA [6] is a keyword-based search engine that translates natural language questions into conjunctive SPARQL queries. In the query pre-processing step (2) shown in figure 2.12, SINA reduces input question queries into representative terms/keywords before lookup in available Semantic Web resources. Similarly to another keyword-based approach **NLP-reduce** [19], this reduction is done by applying NLP methods such as tokenization, stop-word removal and stemming/lemmatization to the query, so that resources can easily be matched to the resulting keywords. In the segment validation step (3), the keywords are segmented into groups of varying length, fitting them to available resources. For instance, the keywords "Fight" and "Club" would be grouped into "Fight Club" and be recognized as the movie. (4) In the resource retrieval step, the fitted keywords are then string matched to RDF labels in the resources. In the disambiguation step (5), the appropriate subset of resources is retrieved by ranking. (6) The SPARQL query is generated from the graph-structure present in the database for the selected resources.

Contribution

SINA provides a novel approach for determining the most suitable resources for a user-supplied query from different datasets by employing a Hidden Markov Model[4] with different distribution functions.

SINA also supplied a novel method for construction SPARQL queries by leveraging the linking structure of graph databases using a link traversal method for creating a connected graph. However this approach also constraints the engine to only be able to answer conjunctive queries resulting in a connected sub-graph.

Evaluation

SINA evaluate the accuracy of their approach using performance measures such as precision, recall and F-measure on three interlinked datasets as well as DBPedia.

[4]Hidden Markov Models are a class of probabilistic graphical model that lets us predict sequences of unknown variables from a set of observed variables

They also studied the runtime of their approach by implementing both a sequential and a parallel implementation of the system.

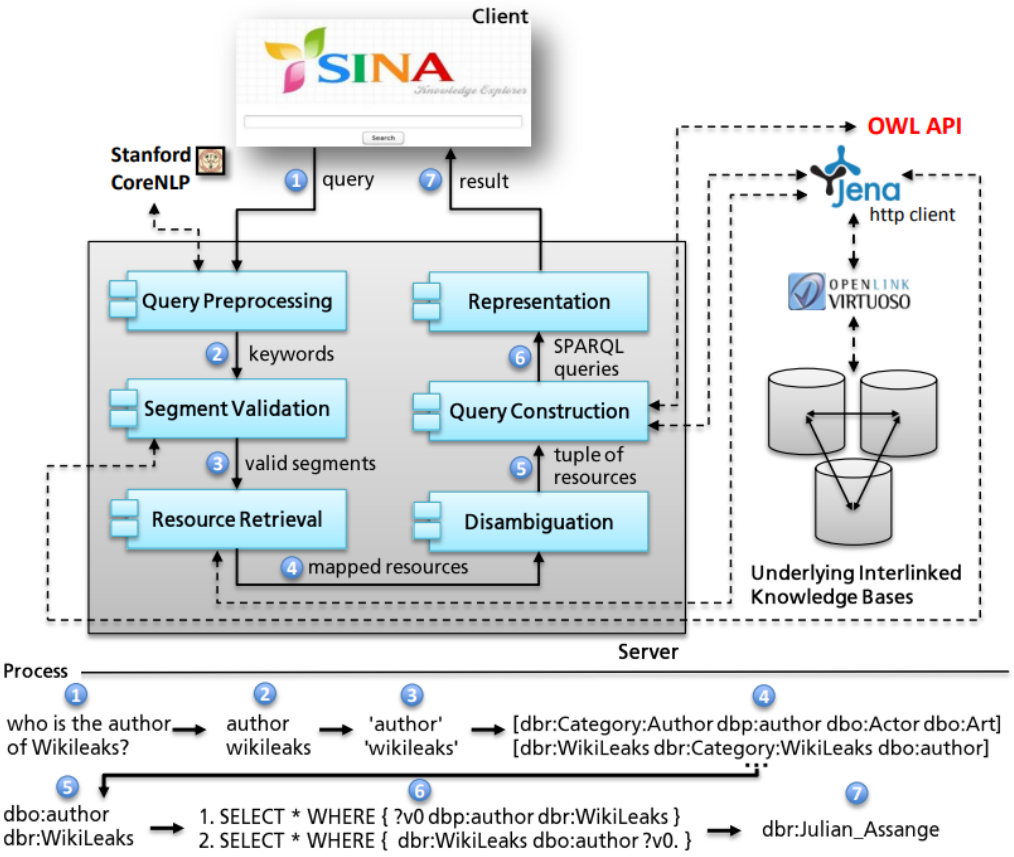


Figure 2.9: Architecture of the SINA search engine. Source:[6]

2.2.2 Athena

Athena [15] is a parsing-based search engine for relational databases, which handles sentence input queries. Athena is ontology-driven which means that it relies on mappings between a domain ontology and the relational database it uses. To be able to mitigate linguistic variation it employs a set of synonyms to be associated with each element in the given ontology. To be able to convert an input question into SQL, an intermediate Ontology Query Language (OQL) is used before translating input to SQL.

Firstly Athena uses an ontology evidence annotator which maps the input query to ontology elements present in the graph. These types of matches are metadata, synonyms and variations of entities, temporal expressions, numerical expressions and dependencies. Secondly Athena creates a ranked list of the interpretations which are ontology elements provided in the first step. The third step is to generate the intermediate query using OQL which is able to express aggregation functions, unions and nested sub-queries. This is done by supporting trigger words in the user query such as "by" for *Group By* function in SQL or *Order By* clause which is triggered by words such as "*least*", "*most*", "*ordered by*" and more. The generated OQL is then used by a translation algorithm which performs all the execution tasks described in OQL for SQL. This includes executing unions of individual OQL queries, performing attribute and join condition transformation. A query that consists of directly translatable attributes and join conditions can be directly translated into a SQL query. The reason for this is that the only database tables accessed are the ones that correspond to the concepts present in the sets of attributes and join conditions.

Contribution

Athena is the first ontology-based NLQ engine for relational databases. Athena provides a novel two-stage approach that translates the input into an intermediate ontology query language before translating to SQL. Athena can also handle more complex queries that requires aggregation functions, unions and nesting.

Evaluation

Athena evaluate the accuracy of their approach using performance measures such as precision, recall and mean reciprocal rank on three different workloads of different data types; geographical, financial and academic.

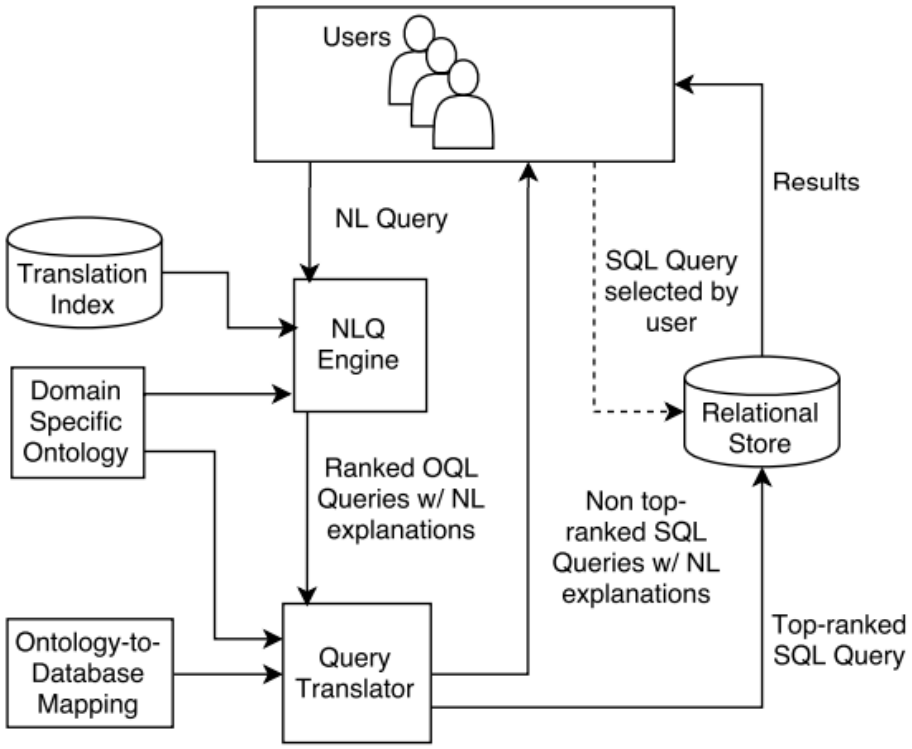


Figure 2.10: Architecture of the Athena search engine

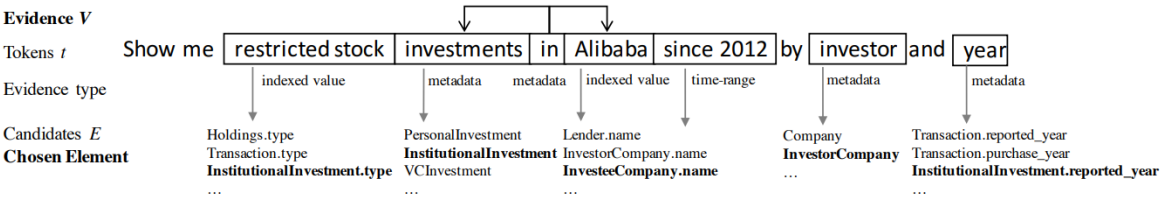


Figure 2.11: Example use of the Athena Ontology Evidence Annotator. Source:[15]

2.2.3 FREyA

FREyA [20] is a parsing-based search engine for automatic SPARQL generation that uses knowledge encoded in the ontologies as the primary source for understanding the user's question and the syntactic parsing as a secondary source to provide more precise answer. It does so by annotating the user input with ontology concepts[5] it is able to recognize by heuristic rules, and engages in clarification dialogues with the user whenever there are ambiguous concepts. FREyA translates a natural language question into a set of ontology concepts by combining the syntactic parsing with an ontology reasoner in order to identify the user's information need correctly. Potential ontology concepts (POC) are identified by the syntactic parse tree generated by the Stanford Parser tool, a precursor to the CoreNLP tool [14] and ranked by string similarity to suggestion including synonym detection by Wordnet [21], which is a synonym lexical resource.

Contribution

FREyA contributes with a model that can take advantage of user feedback to disambiguate concepts and find the right answer. In addition, the user feedback is saved to improve FREyA's performance over time. The drawback is that few queries can be answered without any clarification dialogues.

Evaluation

FREyA is evaluated using the performance measures precision and recall. They also evaluated the performance of their ranking algorithm using mean reciprocal rank. They tested the system on a geography dataset from Mooney containing 250 questions .

[5] Ontology concepts includes all ontology elements:classes, instances, properties and literals

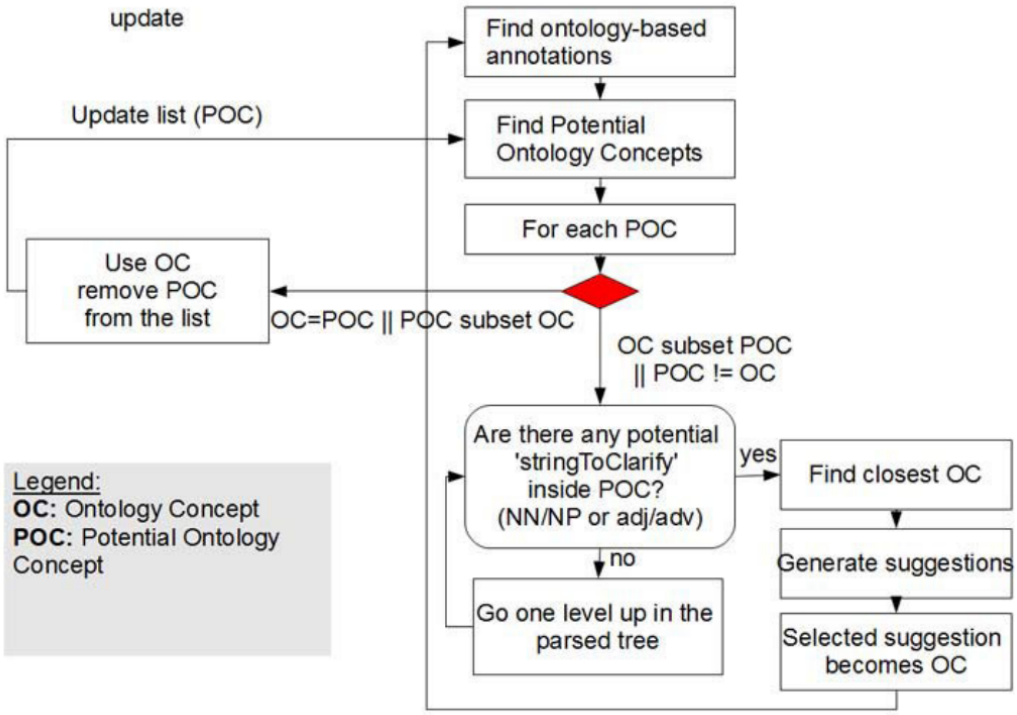


Figure 2.12: Freya: Validation process of potential ontology concepts through user interaction. Source:[20]

2.2.4 Querix

Querix [4] is a parsing-based approach that lets users enter questions in natural language to query an ontology. Similarly like FREyA [20], if the engine recognizes ambiguous input tokens, it then uses clarification dialogues querying the user. Querix also uses a syntax parse tree to extract the sequences of words and their part of speech tags to create a query skeleton. Querix also applies Wordnet [21] to supply synonyms to the query skeleton. The query skeleton is then used to match RDF triple patterns in the ontology to possible subject - predicate - object triples found in the user input query or its synonyms.

Contribution

Querix contributes with a simple model that is easy to use and portable as it does not need any adaptation for new ontologies. However the simplicity also reduces the number of questions that can be answered as it has to include the predefined syntax.

Evaluation

Querix is evaluated using the performance measures precision and recall. They tested the system on a knowledge base containing geographical information and ran 215 queries.

2.2.5 Aqqu

Aqqu [8] is a template-based Question Answering System over Freebase[6] which focuses on "structurally simple" questions. Such questions contains a few knowledge base entities and relations where the natural language terms needs to be correctly mapped to entities and relations(properties) present in the Knowledge Base. Aqqu initially identifies the entities present in query by mapping mentions to entities in the Freebase database. Aqqu delays the disambiguation step, and the result of the entity identification step is a set of possibly overlapping entities mentions with attached confidence scores. Next, Aqqu matches the query question to a template. The templates are predefined sub-graphs which in the end constitute the sub-graph pattern in the resulting SPARQL query. Candidate predicates and objects are found by executing ambiguous SPARQL queries where only the entity is known, and placeholder variables are put in place for the rest of the template. In the relation matching step the ambiguous query candidates are scored by having their predicates matched to relations mentioned in the question. They do this by string matching the name or description of the relation in the KB. Each of these matches has a confidence score attached. Last step is the ranking step, where each query candidate is enriched with information about which entities and relations match which parts of the question. Candidates that cover most of the question is intuitively ranked as the best.

Contribution

Aqqu contributes with a simple template model that performs surprisingly well. By doing ranking in the final step, Aqqu has the benefit of jointly disambiguating entities and relations at the same time. This way, a candidate can have a weak entity match, but a strong relation match and still be highly ranked as a possible candidate.

Evaluation

Aqqu is evaluated using the performance measures accuracy and average F1. They tested the system on Freebase KB an evaluated using two established benchmarks: Free917 containing 267 questions and WebQuestions containing 2032 questions. These benchmarks contains a set of question and Freebase answer entity pairs as their format.

[6] Precursor Knowledge Base to Wikidata

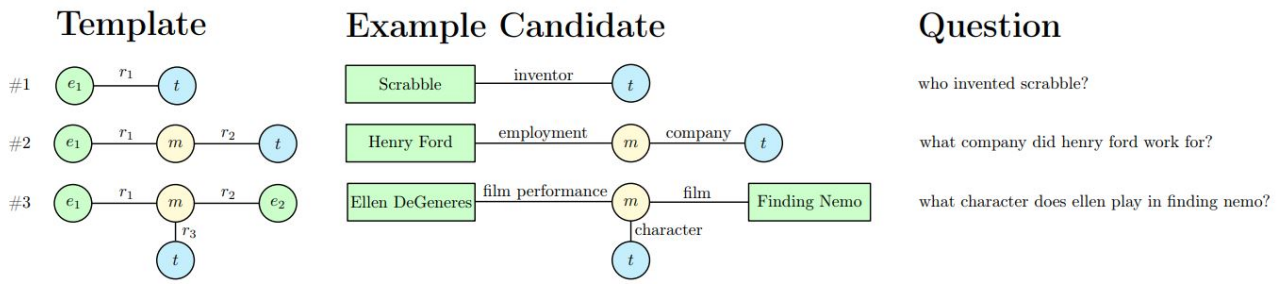


Figure 2.13: Aqqu: Query templates and example candidates with corresponding questions. A query template can consist of entity placeholders e, relation placeholder r, and intermediate object m and the answer node t. Source:[8]

Chapter 3

Method

This chapter presents the research methods applied in this project, the limitations to the chosen methods, the chosen approach to solve the research questions, research goal and hypothesis.

3.1 Research Methodology

Finding Related Work

The first step of the research methodology was a phase of familiarization with the domain, by finding previous work in the area by keyword search on Google Scholar[1] and DLBP[2] as well as look-ups to unknown concepts or methods on Google. General keywords used were for instance "Question Answering", "Natural language interfaces", "Natural Language interface Knowledge Graph", "Question Answering Knowledge Base". Comparative studies were useful resources to understand what domain specific state of the art approaches were available. In addition, look-ups in the references were applied to quickly gain access to relevant work in the nested web of references. NTNU-Open[3] contains other master's theses and PhD theses that were inspiring and beneficial to read through.

Developing a Question Answering System

To be able to answer the research questions, hypothesis and goal described in section 1.2, a Question Answering system over Wikidata was developed with the content of section 1.2 in mind.

[1] Google Scholar Link: https://scholar.google.com/
[2] DLBP is a computer science biblography: https://dblp.org/
[3] NTNU-Open available at: https://ntnuopen.ntnu.no/ntnu-xmlui/

3.2 Approach

When deciding the chosen approach, a several things was taken into consideration. The available Norwegian specific NLP tools and dataset resources were browsed. After realizing what resources and dataset were available it was also a process of fitting the tools to the different approaches and related work presented in chapter 2, to shape a method that could be compatible for Norwegian Questioning Answering over Wikidata. It was also decided to create entity and relation indexes to link question entity and relation mentions to KB items as no such dataset existed for Norwegian, which is vital for answering natural language queries using a Knowledge Base. This predicament in particular had a lasting effect on the chosen approach.

Our approach for creating the Question Answering System was based on the approach Aqqu [8] explained in section 2.2.5 with a few tweaks:

- focuses on the simple query template with only 1 RDF triple required to answer a question, skipping the template matching routine
- creating a Norwegian entity index as well as a Norwegian relation index for faster look-ups
- replaces the learning-based ranking approach with a heuristic feature-based ranking

This approach is mainly reliant on the Norwegian entity- and relation aliases present in the KB (as well as the KB itself), in order to answer the user question. In addition the approach uses the SpaCy library with Norwegian support for Norwegian specific pre-processing of the question.

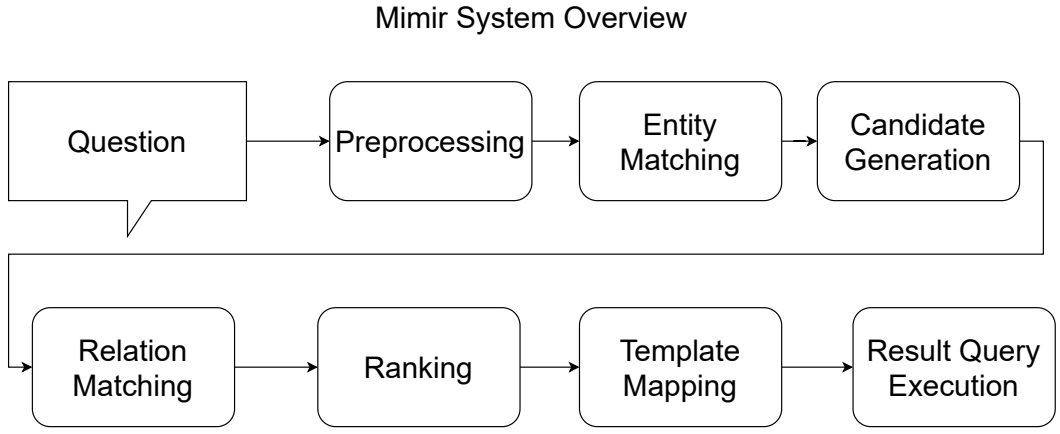


Figure 3.1: Overview of the Mímir end-to-end pipeline.

3.2.1 Pre-processing the Question

Tokenization

Tokenization is the process of splitting each of the terms in the questions are split into tokens. The tokens are single words split by white-space, and looks similarly to the unigrams present in table 3.1. Tokenization is done to enable further processing in later stages, such as **PoS Tagging** and **Lemmatization**.

N-gram Generation

To be able to match an alias of varying word length to an entity in the KB, the question is split into **N-grams**. These N-grams are used to map entity mentions in the question to entity aliases of varying term length present in the KB.

N-grams

Sentence	Hvem er forfatteren av naiv super
Unigrams	'hvem', 'er', 'forfatteren', 'av', 'naiv', 'super'
Bigrams	'hvem er', 'er forfatteren', 'forfatteren av', 'av naiv', **'naiv super'**
Trigrams	'hvem er forfatteren', 'er forfatteren av', 'forfatteren av naiv', 'av naiv super'
Four-grams	'hvem er forfatteren av', 'er forfatteren av naiv', 'forfatteren av naiv super'
Five-grams	'hvem er forfatteren av naiv', 'er forfatteren av naiv super'

Table 3.1: Table showing different N-grams of a Norwegian question sentence which are used to link entity mentions to entities in the KB. Note that it is the Bigram **'naiv super'** which is the correct entity mention for this question.

Lemmatization

Lemmatization is the process of converting a token into its canonical form. For instance a conjugated word is converted into its Infinitive. This is done to increase the number of possibly matching relations in the relation step described in subsection 3.2.4.

POS Tagging

POS Tagging is useful as it tells us information about the grammatical structure of each term. Each of the question terms is annotated with a POS-tag, classifying terms as nouns, verbs, adjectives and more. This is extremely useful especially in the relation matching step.

3.2.2 Entity Matching

Norwegian Alias Entity Index

The entity index is the result of a SPARQL query that retrieves all Norwegian aliases and their linked entity URI. The actual SPARQL query can be found in the implementation chapter 4.3.2. The query resulted in an entity-alias index of over 7 million aliases and their connected entities. In particular for each entity we retrieve fields which can be used as Norwegian aliases. Specifically these consist of relations that return literals for:

- **'Familiy name'** - <http://www.wikidata.org/prop/direct/P734>
- **'Short name'** - <http://www.wikidata.org/prop/direct/P1813>
- **'Birth name'** - <http://www.wikidata.org/prop/direct/P1477>
- **'Nickname'** - <http://www.wikidata.org/prop/direct/P1449>
- **'Pseudonym'** - <http://www.wikidata.org/prop/direct/P742>
- **'Label'** - <http://www.w3.org/2004/02/skos/core#label>
- **'Alternative Labels'** - <http://www.w3.org/2004/02/skos/core#altLabel>

The literals had all their uppercase characters lowered and "unidecoded" to ensure more matches to question terms where such things are not differentiated by the user. The dataset was then grouped and aggregated by alias resulting in a dataset of distinct aliases and their connected Wikidata entity URI where the aggregated dataset consist of 3 million rows containing aliases and their set of connected entities.

Entity Identification and Linking

To be able to recognize and link entities mentioned in the question, a mapping step between the question terms and the entities in the KB is required. This is done by looking up the inverted entity index of all Norwegian aliases for entities and matching them to the N-grams of the input question. The results of this step is a set of possibly overlapping entities matched from the different N-grams, as any alias can have several connected entities. Each N-gram entity match is also marked with a N-gram score, based on the number of words in the N-gram matched to the entity.

N-gram matched Entity Alias Index

Alias	Entity Links	Description
'hvem'	http://www.wikidata.org/entity/Q504961	A 1927 film where 'hvem' is the Norwegian name.
'super'	http://www.wikidata.org/entity/Q66116090	American comedian Glenn Super
'super'	http://www.wikidata.org/entity/Q37318743	Refers to the family name Super
'super'	http://www.wikidata.org/entity/Q3027202	French musician named Didier Super
'naiv super'	**http://www.wikidata.org/entity/Q1767831**	**A novel by Erlend Loe**
...	...	...

Table 3.2: Table showing example linked entity aliases to N-grams entity mentions in the question. The bold row is the correct match for retrieving the answer.

3.2.3 Triple Candidate Generation

We use the simple template pattern described as **Template 1** in Aqqu [8] which consists of a single *Subject - Predicate - Object triple* as our template for generating candidate triple statements. Because we only answer simple queries on this template form, the template matching step present in Aqqu is omitted.

Let E be the set of all KB entities matched to a N-gram of the question as described in previous subsection. For each entity $e \in E$, we generate all predicates and objects related to that entity by executing a single SPARQL query for each e. This gives us the full list of relation triples that any linked entity has.

- For each entity $e \in E$ Query$(e, ?P, ?O)$,
  where $?P$ is an ambiguous predicate relation variable and
  $?O$ is an ambiguous object variable.

The result of this step is the ambiguous triple candidates which constitute the set of all query candidates for the question.

3.2.4 Relation Matching

Norwegian Alias Relation Index

The relation index is the result of a SPARQL Query that retrieves all Norwegian aliases and their linked relation URI. The actual SPARQL query can be found in the implementation chapter 4.3.3. The query resulted in an alias - relation index of over 11 000 aliases and their connected property relation. In particular for each relation we retrieve labels which can be used as Norwegian aliases.

- **'Label'** - <http://www.w3.org/2004/02/skos/core#label>
- **'Alternative labels'** - <http://www.w3.org/2004/02/skos/core#altLabel>

The dataset was then grouped and aggregated by alias resulting in an inverted index dataset of distinct aliases and their connected Wikidata property URI where the aggregated dataset consist of 5000+ rows containing aliases and their set of connected relation properties.

Relation Identification and Linking

The query candidates from the last step still miss important information about which relations were actually mentioned and asked for in the query. By matching the relation mentions in the question to any of the Norwegian aliases of the predicates present in the KB, it is possible to rank triple candidate statements (from the last step) with the connected relations present in the question. Similarly to the inverted entity index, this is done by using the inverted relation index which consists of all predicates with Norwegian aliases present in the KB. For each term in the question that are not already matched to an entity, and is marked as a relation PoS tag (verbs, adjectives, numbers or nouns), we match the *lemmatized* question terms to relation aliases in the inverted relation index. The matched relation URI is then used to attach relation scores to the generated triple candidates in the previous step. Each exact relation match was marked with a high relation score on the triple candidate. Substring relation matches are marked with a lower relation score on the triple candidate.

For the Question "*hvem er forfatteren av naiv super?*", the correct linking "*forfatteren*" - lemmatized "*forfatter*" is matched to the predicate URI "*http://www.wikidata.org/prop/P50*" which describes the **Author** relation.

Matched Relation Alias Index

Alias	Relation Links	Match
'forfatter'	http://www.wikidata.org/prop/P50	**Exact Match**
'manusforfatter'	http://www.wikidata.org/prop/P58	Substring Match
'google-scholar forfatter-id'	http://www.wikidata.org/prop/P1960	Substring Match
...	...	...

Table 3.3: Table showing example linked relation aliases to lemmatized relation mentions in the question. The bold row is the highest valued match for retrieving the answer.

3.2.5 Ranking

We have now a collection of triple candidates for the answer query, where each triple has information about how well it matches entities and relations in the question query. The only thing left to do is to rank the triple candidates correctly, by leveraging available features correctly. As a general notion, the candidate covering most words of the question is the best, similarly to the ranking in Aqqu [8]. Features used to rank each triple candidate consists of:

1. Number of entities of the candidate that were matched to words in the question by alias.
2. Number of exact relation matches
3. Number of substring relation matches
4. Entity and Relation Coverage - number of matched entity and relation terms in the candidate divided by number of terms in the question.

The candidate collection is then sorted by the highest combined score, which will constitute the best triple candidate to answer the question.

3.2.6 Template Mapping and Answer SPARQL Execution

The last step is to map the best ranked candidate to a SPARQL query template, where the candidate triple constitutes the sub-graph pattern for getting the correct answer. We return in the template both the wikidata entity identification and label (if it is present in the KB).

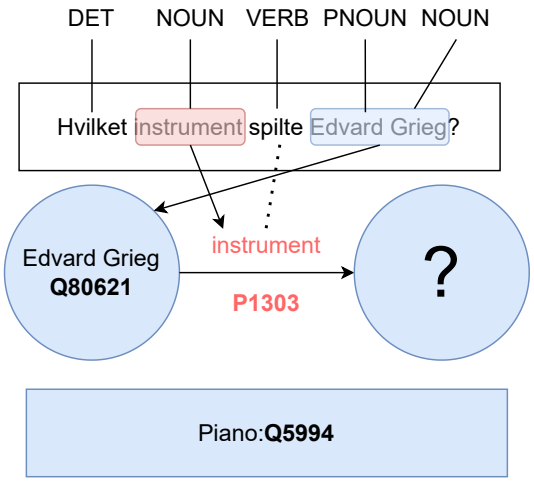


Figure 3.2: Visualization of simple question answering over Wikidata in Norwegian using Mímir approach. The sentence terms are marked with PoS tags. The Entities are marked in blue and the correct relation is marked in red. Note that "spilte" in English "played" is also a valid semantic relation, but does not describe the entity-instrument relation present in the KB.

3.3 Evaluation Methods

To evaluate Mímir, the performance metric accuracy was applied. Accuracy is the fraction of the queries answered with the exact gold answer. The gold answer refers to (in this case) the objective truth/answer to the query, which is present by the gold query which is the correct SPARQL query triple to achieve the correct result. To have a dataset to evaluate question answer pairs, we created a test set of manually translated question answering pairs from Question Answering Benchmarks for Wikidata [22] resulting in a dataset of 20 + question answer pairs. We also created a test set of 20+ question-answer pairs inspired by Simple Question Dataset, with the goal of testing the capabilities of the system.

Chapter 4

Implementation

This chapter gives an overview of the technology, tools and datasets used to implement the methods described in chapter 3.

4.1 Python and Python Libraries

Python has been the programming language used to implement all parts of the system, including the evaluation experiments. Python is a language with a large collection of libraries for different purposes. The ones used in this project are listed below.

SpaCy SpaCy[1] is a natural language toolkit for processing text in natural languages, like English or Norwegian. For Norwegian it is trained on 3 different news datasets, where the largest is a file of 542MB[2]. It includes methods for tokenization, lemmatization and PoS tagging among others, which are used in this project.

Unidecode Unidecode[3] is a library to normalize a text input to its nearest unicode representation. To be able to find all matching entities with special characters such as "*Mímir*", we use this library to normalize the entity mention to its nearest unicode representation, namely "*Mimir*". This is done to account for users that exclude inputting special characters in the question, as well as increasing the number of matches. This is done both in the question preprocessing step and the entity alias indexing step.

SPARQLWrapper SPARQLWrapper[4] library is a python wrapper to easily query SPARQL service endpoints and execute queries from python scripts. SPARQLWrapper helps create query invocation and convert the results into manageable formats.

[1] https://spacy.io/
[2] https://spacy.io/models/nb#nb_core_news_lg
[3] https://pypi.org/project/Unidecode/
[4] https://sparqlwrapper.readthedocs.io/en/latest/main.html

Pandas Pandas[5] is a fast and powerful data analysis and manipulation tool for python, that is used to store the data received from the SPARQL queries executed in this project and process the data. Pandas helps simplify working with multidimensional arrays and matrices, including methods for operating and formatting the large datasets present in the indexation steps and query candidate generation and ranking steps.

4.2 Wikidata SPARQL Endpoint

To be able to query the Wikidata Knowledge Base, we use the available Wikidata SPARQL endpoint[6] which allows us to execute queries towards their database in a somewhat restricted environment. Wikidata has implemented limitations on the amount of query or processing power allowed to do towards this endpoint in a small time window. This is reasonable as it is detrimental for their endpoint to break as a result of large, encumbering queries. Even with these restrictions it works to query for the best ranked candidate and to some extent generate query candidate. However for constructing the inverted indexes for entities amounting to over **7 million Norwegian aliases**, querying the Wikidata endpoint directly for this information was not possible.

4.3 Qlever Query Engine and SPARQL Endpoint

QLever [23] is a fast SPARQL Query engine which was relatively easy to implement, given their tutorial available on their Github page[7]. QLever was a possible option for building our own back-end SPARQL endpoint which we could run our extensive queries on. The last step missing was downloading and building a Wikidata dump on top of our QLever implementation. The lastest Wikidata dump was an $>$ 100GB compressed file[8] which took a week to download. However to be able to build such a big query engine, it would require either a powerful computer or a computer cluster, and even then it would take a long time to procure.

The solution was to use the externally available QLever SPARQL endpoint[9] which had already build a Wikidata engine, that we could use to query for the aliases. By implementing a indexing python script, the entity alias index and the relation alias index were created for Norwegian aliases.

4.3.1 Official Written Languages Bokmål and Nynorsk

The Wikidata Knowledge Base contains multilingual support for many languages. In the case of Norwegian there are 2 official languages, and Wikidata supports

[5]https://pandas.pydata.org/
[6]Available Wikidata query demo at: https://query.wikidata.org/
[7]https://github.com/ad-freiburg/qlever
[8]Available at: https://www.wikidata.org/wiki/Wikidata:Database_download
[9]QLever endpoint: https://qlever.cs.uni-freiburg.de/api/wikidata-proxy

both. "Bokmål", the written standard of Norwegian has the most aliases available on Wikidata with >7 million entity - alias pairs, and therefore the highest chance of linking question entities to KB items. "Nynorsk" or in English "New Norwegian" has fewer aliases available at >6 million and is used by a minority of Norwegians. For these reasons the implementation only supports Norwegian Bokmål.

4.3.2 Norwegian Alias Entity Index

Inverted Index Query:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
SELECT ?entity ?alias WHERE {
?entity wdt:P734?/(@nb@wdt:P1813|@nb@wdt:P1477|@nb@wdt:P1449|@nb@wdt:P742|@nb@skos:
    altLabel|@nb@rdfs:label) ?alias .
MINUS {
# Ignoring internal items
?entity wdt:P31/wdt:P279* wd:Q17442446 .
}
}
ORDER BY DESC(?entity) DESC(?alias)
```

This is the SPARQL/QLever query for getting all the entities and their connected Norwegian aliases. Note that "@nb@" is a QLever filtering method, where "@nb" is an RDF label that stands for "Norsk bokmål".To get the aliases for English one can simple swap "@nb" with "@en". For English there exists >98 million Wikidata entity - alias pairs using this query.

4.3.3 Norwegian Alias Relation Index

Inverted index Queries:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX wikibase: <http://wikiba.se/ontology#>
SELECT ?predicate ?alias WHERE {
{
SELECT ?predicate WHERE {
?x ql:has-predicate ?predicate .
}
GROUP BY ?predicate
}
?entity wikibase:claim ?predicate .
OPTIONAL {
?entity @nb@rdfs:label|@nb@skos:altLabel ?alias .
}
}
ORDER BY ASC(?predicate) ASC(?alias)
```

This is the SPARQL/QLever query for getting all the relations and their connected Norwegian aliases in the form of labels or alt labels. For Norwegian Bokmål there exists >11 000 rows of alias - relation pairs. For English there are >25 000 using the same query. Both the Norwegian Alias Entity Index and Norwegian Alias Relation Index are made publicly available at our github page[10].

4.3.4 Answer format

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT DISTINCT ?result ?resultname
WHERE { ?entity_URI ?relation_URI ?result .
OPTIONAL { ?result rdfs:label ?resultname .
FILTER (lang(?resultname) = "nb") . } . }
LIMIT 10
```

The answer is found by executing a SPARQL Query with the highest ranked candidate triple's Entity and Relation URI, which then returns the URI and the Norwegian label for the object in the sub-graph pattern. For results that return a literal like numbers and dates (e.g not other entities) we return only the value, as it does not have a label attached. Note that even though we return top 10 with the LIMIT syntax we only really use the first result.

4.4 Simple Norwegian Questions for Wikidata

To be able to evaluate my Question Answering system, a dataset of 20+ questions were translated from the Question Answering Benchmarks for Wikidata [22], specifically *SimpleQuestions for Wikidata* which consists of 21 957 questions deemed answerable on Wikidata (in English). These questions had the potential issue that of not existing with a Norwegian counterpart given the lack of labels used in the entity and relation index, so some additional checks for answer-ability has to be made.

Similarly for the analysis, a dataset of 20+ questions were created on the same format as [22] to test the capabilities of the system. The questions were created by asking general questions which should exist in the general wikidata corpus for Norwegian, and were deliberately dissimilar as a means to analyse different question arch-types the implementation should be able to answer.

[10]Norwegian Question Answering dataset github: https://github.com/hakon0809/Norwegian-Question-Answering

4.5 Architecture Overview

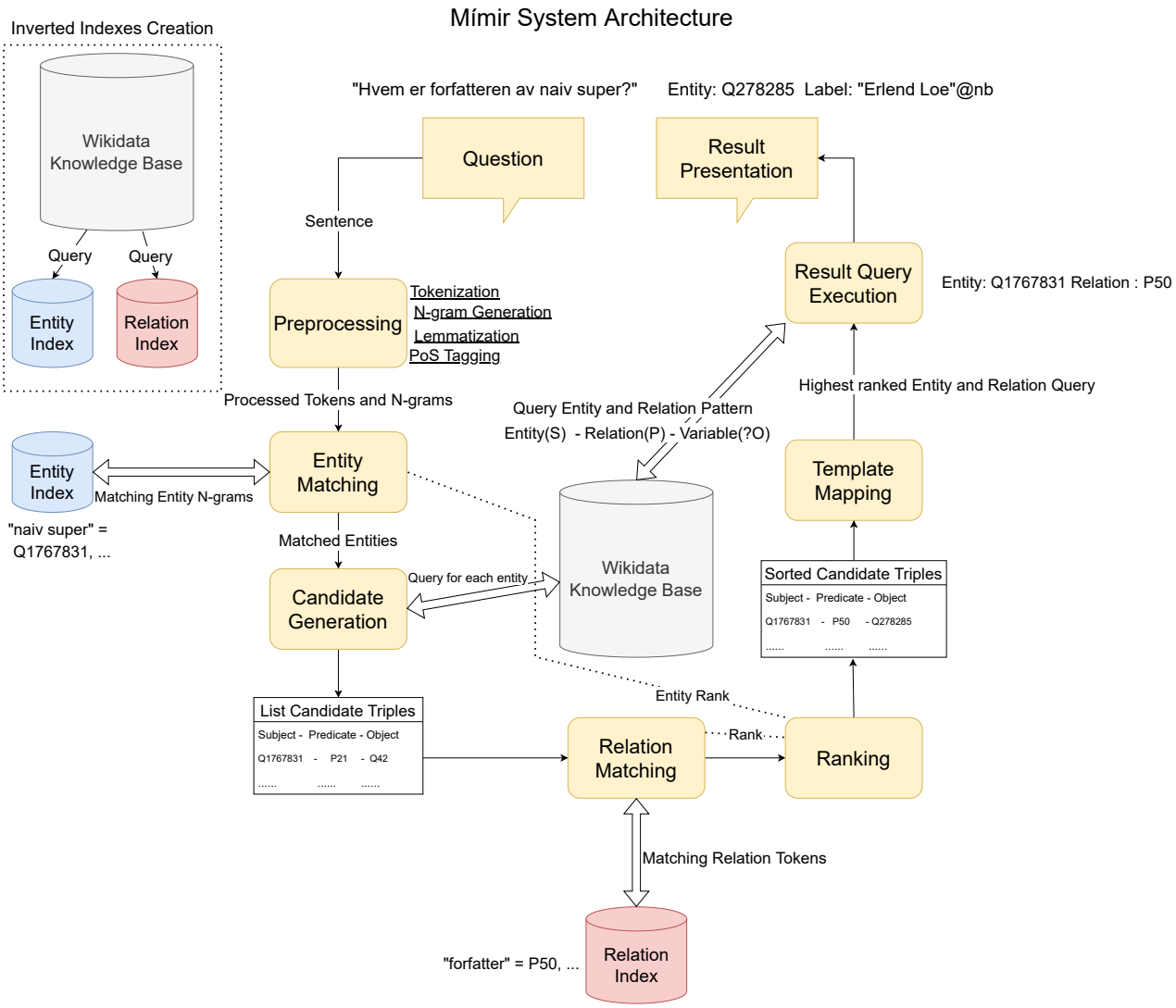


Figure 4.1: Overview of the Mímir implementation. It shows the pipeline of asking a Norwegian simple question "*Hvem er forfatteren av naiv super?*" translated to English "*Who is the author of naiv super?*". The system relies on identifying aliases from the KB and mapping them to terms in the question sentence.

Mímir is only usable by terminal, and prompts the user for a question to be processed in the pipeline described in figure 4.1. Although a online demo was planned, it was ultimately discarded given time limitations.

Chapter 5

Evaluation and Discussion

In this chapter we present the datasets used to setup our testing experiment and review the results gathered through testing and evaluation. The results are then used in the analysis, followed by a discussion about the project as a whole and its limitations.

5.1 Datasets

Our evaluation and experiments done are performed using the dataset available in the Wikidata Knowledge Base by SPARQL Queries to SPARQL endpoints such as the endpoint provided by Wikidata and QLever. In addition we also use the inverted indexes for entities and relations.

For actually testing the system we looked at different datasets. The problem with selecting a testing dataset is that there still does not exist a Norwegian specific question answering testset for knowledge bases - to the best of my knowledge. Even though QALD 9 [1] challenge focuses on provides testing sets for Questioning Answering over Knowledge Bases with multilingual support, the challenge dataset contains support for 11 languages of large amounts of speakers like English, Spanish, and German. QALD challenges are unlikely to cover Norwegian in the near future as has is a rather small amount of speakers. In addition these question answer pairs are tuned for complex SPARQL queries which my simple implementation cannot answer anyways (for now).

Question Answering Benchmarks for Wikidata [22] provides a dataset of simple question-answer pairs for Wikidata named *SimpleQuestions for Wikidata*. *SimpleQuestions for Wikidata* was created from a dataset originally made for Freebase[1]. The Freebase dataset is described in the paper SimpleQuestions [24]. The Freebase SimpleQuestions dataset provided 108 442 questions annotated with a Freebase triple such that one of the acceptable answers to the question is the subject of the triple. *SimpleQuestions for Wikidata* [22] consists of 49 202 questions on which 21 957 are answerable over Wikidata. It is from these simple questions

[1]Freebase is another Knowledge Base.

that we create our Norwegian test set amounting to 25 question-answer pairs. Note that we have to manually translate each question to admissible Norwegian Bokmål, making this a tedious and time-consuming task. It is for this reason that we are limited to translating and creating a small subset of the answerable *SimpleQuestions for Wikidata*. To make the Norwegian testset somewhat representable of the much larger English counterpart, the 25 chosen questions are selected randomly. However, this effort does not make the small subset comparable to the *SimpleQuestions for Wikidata* testset.

In addition we also create a small Mímir analysis testset of Norwegian Questions amounting to 30 question-answer pairs catered to Norwegian, which generally should be answerable by our system. This testset is used to analyse the capabilities and limitations of the implementation based on our knowledge of Norwegian, as well as compare it to the Norwegian translated SimpleQuestions testset.

5.2 Evaluation

We evaluate our implementation and method by the performance metric accuracy. Accuracy is the fraction of questions answered with the correct answer. We only include small evaluation sets as these are self-implemented. The goal of this evaluation is not to compare our implementation to state of the art models or outperform them, but rather show Mímir's potential as a proof of concept and validity as a simple Norwegian Question Answering System over Wikidata.

5.2.1 Results

Testset	Accuracy	Number of Questions
SimpleQuestions Wikidata to Norwegian	48%	25
Mímir Analysis Testset	73%	30

Table 5.1: Table showing accuracy results for the two evaluation testsets used in this project.

Unsurprisingly, the manually translated SimpleQuestions Wikidata testset managed to correctly answer fewer questions than the Mímir analysis testset. This is mainly due to the fact that there exists fewer Norwegian aliases than English, thus having a smaller chance of linking mentions to correct KB items for questions meant for English.

Best Ranked Entity and Relation for each Question

Entity	Entity Label	Relation/Property	Relation Label @nb|@en
Q10280325	Fausto Fawscett	P21	Kjønn | Gender
Q66832	Franz Roh	P27	Nasjonalitet | Nationality
Q568123	Bram Stoker's Dracula	P136	Sjanger | Genre
Q168648	Sopron	P421	tidssone | located in time zone
Q3393521	Joaquín Cardiel	P264	platemerke | record label

Answer Object and Questions

Answer Object Entity	Answer Label or Literal @nb|@en	Norwegian Question	English SimpleQuestion
Q6581097	Mann | Male	hvilket kjønn er fausto fawscett?	what is fausto fawcett's gender?
Q183	Tyskland | Germany	hva var nasjonaliteten til franz roh?	What was the nationality of franz roh?
Q1054574	Romantisk Film | Romance Film	hvilken sjanger er filmen dracula?	what kind of movie is dracula?
Q25989	Sentraleuropeisk tid | Central European Time	hvilken tidssone er sopron i?	which time zone is sopron located in?
Q183412	EMI | EMI	hvilket plateselskap er joaquin cardiel signert til?	What label is joaquin cardiel signed to?

Table 5.2: A split table showing 5 example results from the Norwegian translated SimpleQuestions for Wikidata. Note that the first row of the first table-half responds to the first row of the second table-half.

A subset of 5 correctly answered question-answer pairs from the Norwegian SimpleQuestions for wikidata are presented in table 5.2. The first table-half contains the best ranked recognized entity and relation in each question which is then used as the triple pattern to produce the answer entity and label shown in the second table-half.

Best Ranked Entity and Relation for each Question

Entity	Entity Label	Relation/Property	Relation Label @nb|@en
Q30	United States of America	P36	hovedstad | capital
Q66832	naiv super	P50	forfatter | author
Q7251	Alan Turing	P136	morsmål| native language
Q585	Oslo	P1082	folketall | population
Q362	World War II	P580	startdato | start time

Answer Object and Questions

Answer Object Entity	Answer Label or Literal @nb|@en	Norwegian Question	English Equivalent Question
Q61	Washington, D.C.| Washington, D.C.	hva heter hovedstaden i usa?	what is the capital of USA?
Q278285	Erlend Loe | Erlend Loe	hvem er forfatteren av naiv super?	Who is the author of naiv super?
Q1860	Engelsk | English	hva var morsmålet til alan turing?	What was the native language of alan turing?
-	693494	hvor mange innbyggere er det i oslo?	How many people live in oslo?
-	1939-09-01T00:00:00	når var starttidspunktet til andre verdenskrig?	When did World War Two start?

Table 5.3: A split table showing 5 example results from the Mímir analysis testset. Note that the first row of the first table-half responds to the first row of the second table-half.

A subset of 5 correctly answered question-answer pairs from the Norwegian SimpleQuestions for wikidata are presented in table 5.3. This table is on the same format as table 5.2. Notice that the answer does not have to include an entity as shown by the resulting literals in the two last questions: "How many people live in oslo?" which results in the number people that live there, or the datetime answer for the question "When did World War Two start?".

5.3 Analysis

The analysis is done using the self-implemented question answering pairs named Mímir Analysis Testset. We evaluate the results by comparing the generated SPARQL triple query pattern to the gold query. The analysis also includes comparison of the gold answer (actual result) when analysing Mímir for capabilities. When looking at the arch-types of questions which Mímir is able to answer we get this criteria list:

- Requires the question to have a single gold answer. Lists of equally scored results, have varying answerability.
- Requires the system to recognize at least one Norwegian entity and one Norwegian relation to be able to answer a question. Candidate query triple contains only one variable: S - P - ?O, where ?O is the unknown variable.
- Requires the question to be simple in nature. Complex queries requiring multiple sub-graph triples are not answerable.

5.3.1 Error Analysis

Missing Norwegian relation aliases in Wikidata

when asking for "hvem er statsministeren i norge?" - in English "who is the prime minister of Norway?" we get the correct response "Jonas Gahr Støre", where prime minister is synonymous to the relation "head of government". The relation is also time dependant in the sense that the information is updated for current head of government, guaranteeing correct information without having to consider this on our implementation side. Head of government is also synonymous to president, however Mímir fails to answer the question "who is the president of USA?" as the relation alias president is not available in Norwegian, even though it should be. This is one of multiple examples of missing Norwegian aliases in the Knowledge Base. Interestingly, it is then possible to ask "Who is the prime minister of USA?" and get the correct answer "Joe Biden", even though the question is formulated a bit weird.

Nuanced Incorrect Data

Note that when looking for a head of government type relation, like in the previous example, "ordfører" or "minister" in english is classified as a synonym to the "head of government" relation which is not true. However when asking for head of government in a municipality, this is partially true as the minister or "ordfører" in the municipality holds the highest seat of governance for that particular entity.

False Positives and Ambiguity

When asking the question "hva er det offisielle språket i USA?", we get the correct answer "Engelsk", but the connected subject entity is an American Album with the

connected relation being the work's language "verkets språk" which also has the misguiding alias "språk", making our model ranking this relation on equal ground to the official language of country relation.

Norwegian Letters and Ambiguity

As a direct result to the "unidecoding" of the query as well as the indexed aliases, the Norwegian letters "æ,ø,å" are decoded to their nearest unicode representation, "ae, o, a". This increases ambiguity as for instance verbs such as "så" meaning *saw* and "sa" meaning *said*, making it harder to resolve ambiguity for questions including overlapping ambiguous relations.

Performance Issues

When asking questions involving the director George Lucas, the amount of entities with alias "George" are so many that the candidate generation execution time is too large, making the query fail - or take unreasonably long time to finish.

Missing Common Relation Labels

Even though there exists many relations with at least one label, the cases with just one are often very formal and not necessarly compatible with common natural language. For instance the relation "place of birth" has only one Norwegian alias namely "fødested" which generally is not used in common speech or written form. Usually when asking for the place of birth of a person, one would normally use the terms "born" or "birth place" which are present aliases for English, but not even the term "født" is present for Norwegian. This makes matching "født" to place of birth relation difficult for Mímir.

Ambigious Relations

When asking the question "Where did Calvin Leavy die?", in Norwegian "Hvor døde Calvin Leavy?", "death" refers both to "dødsdato" - "date of death" and "dødssted" - "place of death". This makes Mímir sometimes return a datetime answer instead of a location.

5.4 Discussion

5.4.1 RDF Graph Data Structure and Norwegian

It is interesting to see how the RDF data model structure allows us to answer queries by linking the correct entity- and relation mention and solving the answer by looking at the single unknown object variable in the graph pattern, similar to an algebraic expression. it is even more interesting how Knowledge Bases with multi-linguistic support can use the RDF filtering method on specific languages to give us language specific answers. For relational databases based on SQL, multi-linguistic support is not possible. For instance, Athena as introduced in 2.2.2 has to convert the question into a complex intermediate ontology query language to be able to leverage the graph data structure for answering questions.

5.4.2 Norwegian Methods

Even though my method was based on the Aqqu approach given the reasons given in the chapter 3 and have been proved to function correctly, any of the approaches presented in the background chapter 2 should theoretically be possible to apply and be made to work with Norwegian. The only conundrum are the methods that are reliant on other language specific datasets or NLP resources that are unavailable or experimental for Norwegian, and the chosen methods should cross-examined with the NLP tools and resources available introduced in section 2.1.6.

Our method was simple in the sense that it only can answer simple Norwegian questions over Wikidata which is quite limiting for the questions it can answer. It is possible to create more advanced Norwegian methods by including the complex query patterns and the template matching routine present in Aqqu or methods from approaches for complex question answering.

5.4.3 Available Norwegian NLP Tools

Another limiting factor is the available Norwegian NLP Tools. Even though there exists libraries like SpaCy which provides quality pre-processing components for Norwegian which satisfies most of our NLP needs, there are only experimental Norwegian named entity recognition available or named entity recognition trained on a much smaller dataset than what is expected for a general Knowledge Base encompassing all general knowledge. In addition, these are not able to link mentions to KB items on their own - which is why we built the alias indexes.

5.4.4 Limitations of KB Datasets

Despite the fact that 7 million Norwegian Bokmål entity aliases are present in Wikidata, this is still a rather small amount considering that there exists 98 million for English. In this context the Norwegian Bokmål aliases amounts to under 10%

of what is available for English. The Norwegian dataset needs to be expanded massively to be able to be comparable to English.

The same can also be said for the Norwegian relation aliases. There exists 11 000 Norwegian aliases for Wikidata relations compared to the 25 000 available English aliases. This is a gap where Norwegian aliases amounts to 44% of what is available for English. This gap is not as large as the one for entity aliases, but still substantial and harmful to the performance of our model.

5.4.5 Limitations of Testsets

The our 2 testsets of 25 and 30 question - answer pairs are small compared to the simpleQuestions for Wikidata testset amounting to >22 000 answerable Wikidata question-answer pairs. These are small as a result of having to translate the English question - answer pairs individually and manually. This makes our testsets incomparable to simpleQuestions for Wikidata. Sizable Norwegian testsets also needs to be developed to be able to compare Norwegian approaches to simpleQuestions for Wikidata and by extension to other state of the art approaches for English. While it is possible to use translators like Google Translate - these often return ambiguous results.

5.4.6 Findings

After collecting the results from the accuracy testsets, the research questions from section 1.2 were re-examined. The qualitative results from the evaluation supported the three research questions. However to state anything outside of the research questions, more testing is required and a larger testset needs to be developed.

Research Question 1 *How can the data present in the Semantic Web Knowledge Base be used to answer natural language queries in Norwegian?*

Findings By leveraging the aliases existing in the KB for entities and relations, and using them in our method in the form of inverted indexes, we have showed in the evaluation results that this is indeed possible. The low accuracy for the English translated subset does not invalidate the approach because of the reduced amount of aliases available in Norwegian, compared to English. The Mímir analysis testset which is primed towards Norwegian answerable questions shows that the method is applicable for answering Norwegian questions.

Research Question 2 *What challenges are present to answer user queries in Norwegian over a Knowledge Base?*

Findings The challenges discovered are threefold: the limited available Norwegian semantic web data, limited Norwegian specific NLP tools, and limited testsets. The results and analysis done shows that there are challenges regarding the limited available Norwegian data in Wikidata and negatively impact our system's performance. The challenges to Norwegian NLP tools available was discovered when browsing for applicable tools for our system. It was discovered that no Norwegian specific test set for simple question answering over Knowledge Base exists, and this had to be manually created. It was also discovered challenges in the analysis regarding performance issues for generating large amounts of matching query candidates which is not scalable.

Research Question 3 *How can we evaluate the performance of our Norwegian Question Answering system?*

Findings We have showed how we can evaluate the performance of our Norwegian answering system by creating 2 Norwegian testsets measuring accuracy. The Norwegian SimpleQuestions manually translated from the renowned English dataset and the Mímir analysis testset created manually for analysing the approach, resulting in a suitable testset for data available in Norwegian Wikidata. However further development of the testsets are required to compare them to English, given their size.

Chapter 6

Conclusion

This chapter summarizes the project and work done with a conclusion, and defines possible paths for future work.

6.1 Conclusion

Through a survey of related work in the domain, it was found that recent research in question answering systems over Knowledge Bases are moving towards multilinguistic support for solving both simple and complex questions, with extensive testing through benchmarks such as [1]. In addition a Norwegian specific question answering system over Knowledge Base has not yet implemented and seemed like a interesting project to undertake.

In this thesis a template-based approach for Norwegian Question Answering system over Wikidata is proposed, combining sub-problems such as entity linking, question answering and natural language interface creation. By leveraging the vast amount of data available in the Semantic Web - especially Norwegian synonyms for entities and relations in Semantic Web Knowledge Bases, The Norwegian NLP tools and resources available, and applying existing NLP approaches, we can answer simple Norwegian questions over such a Knowledge Base. We contribute with the implementation of the Question Answering System Mímir itself, the Norwegian specific challenges discovered in this project, the inverted Norwegian indexes available for download[1] and the small testsets for further development of Norwegian specific Question Answering Systems.

[1]Github page for the Norwegian indexes, and the Norwegian question-answer pair datasets: https://github.com/hakon0809/Norwegian-Question-Answering

6.2 Further Work

6.2.1 User Feedback

By combining the question answering system with user feedback features similar to the ones available in Querix and FREyA can solve many of the ambiguity problems faced in Norwegian Question Answering over Knowledge Bases through clarifying user dialogues.

6.2.2 Complex Norwegian Question Answering

It is possible to expand our approach to involve answering complex questions by applying the complex template patterns present in Aqqu. It is also possible to apply other advanced NLI approaches like the ones presented in related work, on top of the Norwegian inverted indexes we have created for Wikidata. By doing so a state of the art Norwegian Answering System for complex question answering over Wikidata is achievable.

6.2.3 Norwegian Testsets

To be able to compare Norwegian systems to other state of the art English models, larger Norwegian testsets needs to be developed. Language translation tools like Google Translate can be used to translate questions fast and automatically, however problems are to be expected around ambiguous translation or erroneous translations.

6.2.4 Norwegian Knowledge Base Expansion

Knowledge Bases such as Wikidata needs to be actively expanded and developed with more items made available in Norwegian. Given the discovery of the large gap between available Norwegian semantic entity data compared to English on the fraction of 1/10. Furthermore Norwegian synonym corpuses similar to Wordnet [21] for Norwegian can be implemented and combined with Knowledge Bases to provide even more data.

6.2.5 Norwegian Entity recognition and Linking tools

Norwegian specific entity recognition and linking tools can be developed to accurately solve ambiguity problems with semantic data recognition and linking, by creating high performing ranking routines based on approaches similar to TAGME or ELR.

Bibliography

[1] A. Perevalov, D. Diefenbach, R. Usbeck and A. Both, 'Qald-9-plus: A multilingual dataset for question answering over dbpedia and wikidata translated by native speakers,' in *2022 IEEE 16th International Conference on Semantic Computing (ICSC)*, IEEE, 2022, pp. 229–234.

[2] 'Norsk i hundre! norsk som nasjonalspråk i globaliseringens tidsalder: Et forslag til strategi.,' Norwegian Language Council, 2005. [Online]. Available: https://www.sprakradet.no/upload/9832/norsk_i_hundre.pdf.

[3] L. Blunschi, C. Jossen, D. Kossman, M. Mori and K. Stockinger, 'Soda: Generating sql for business users,' *arXiv preprint arXiv:1207.0134*, 2012.

[4] E. Kaufmann, A. Bernstein and R. Zumstein, 'Querix: A natural language interface to query ontologies based on clarification dialogs,' in *5th international semantic web conference (ISWC 2006)*, Citeseer, 2006, pp. 980–981.

[5] F. Li and H. V. Jagadish, 'Nalir: An interactive natural language interface for querying relational databases,' in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '14, Snowbird, Utah, USA: Association for Computing Machinery, 2014, pp. 709–712, ISBN: 9781450323765. DOI: 10.1145/2588555.2594519. [Online]. Available: https://doi.org/10.1145/2588555.2594519.

[6] S. Shekarpour, E. Marx, A.-C. Ngonga Ngomo and S. Auer, 'Sina: Semantic interpretation of user queries for question answering on interlinked data,' *Journal of Web Semantics*, vol. 30, pp. 39–51, 2015, Semantic Search, ISSN: 1570-8268. DOI: https://doi.org/10.1016/j.websem.2014.06.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570826814000468.

[7] D. Song, F. Schilder, C. Smiley, C. Brew, T. Zielund, H. Bretz, R. Martin, C. Dale, J. Duprey, T. Miller *et al.*, 'Tr discover: A natural language interface for querying and analyzing interlinked datasets,' in *International Semantic Web Conference*, Springer, 2015, pp. 21–37.

[8] H. Bast and E. Haussmann, 'More accurate question answering on freebase,' in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, ser. CIKM '15, Melbourne, Australia: Association for Computing Machinery, 2015, pp. 1431–1440, ISBN: 9781450337946.

DOI: 10.1145/2806416.2806472. [Online]. Available: https://doi.org/10.1145/2806416.2806472.

[9] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer and C. Bizer, 'Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia,' *Semantic Web Journal*, vol. 6, Jan. 2014. DOI: 10.3233/SW-140134.

[10] D. Vrandečić and M. Krötzsch, 'Wikidata: A free collaborative knowledgebase,' *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[11] P. Ferragina and U. Scaiella, 'Tagme: On-the-fly annotation of short text fragments (by wikipedia entities),' in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 1625–1628.

[12] F. Hasibi, K. Balog and S. E. Bratsberg, 'Exploiting entity linking in queries for entity retrieval,' in *Proceedings of the 2016 acm international conference on the theory of information retrieval*, 2016, pp. 209–218.

[13] S. Bird and E. Loper, 'Nltk: The natural language toolkit,' Association for Computational Linguistics, 2004.

[14] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard and D. McClosky, 'The stanford corenlp natural language processing toolkit,' in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.

[15] D. Saha, A. Floratou, K. Sankaranarayanan, U. F. Minhas, A. Mittal and F. Özcan, 'Athena: An ontology-driven system for natural language querying over relational data stores,' *Proceedings of the VLDB Endowment*, vol. 9, pp. 1209–1220, Aug. 2016. DOI: 10.14778/2994509.2994536.

[16] K. Affolter, K. Stockinger and A. Bernstein, 'A comparative survey of recent natural language interfaces for databases,' *The VLDB Journal*, vol. 28, no. 5, pp. 793–819, 2019.

[17] S. Schrage, 'Ontology-based transformation of natural language queries into SPARQL queries by evolutionary algorithms,' Ph.D. dissertation, University of Göttingen, Germany, 2022. [Online]. Available: https://nbn-resolving.org/urn:nbn:de:gbv:7-21.11130/00-1735-0000-0008-59FC-9-8.

[18] P. E. Solberg, A. Skjærholt, L. Øvrelid, K. Hagen and J. B. Johannessen, 'The norwegian dependency treebank,' 2014.

[19] E. Kaufmann, A. Bernstein and L. Fischer, 'Nlp-reduce: A naive but domain-independent natural language interface for querying ontologies,' in *4th European Semantic Web Conference ESWC*, Springer Berlin, 2007, pp. 1–2.

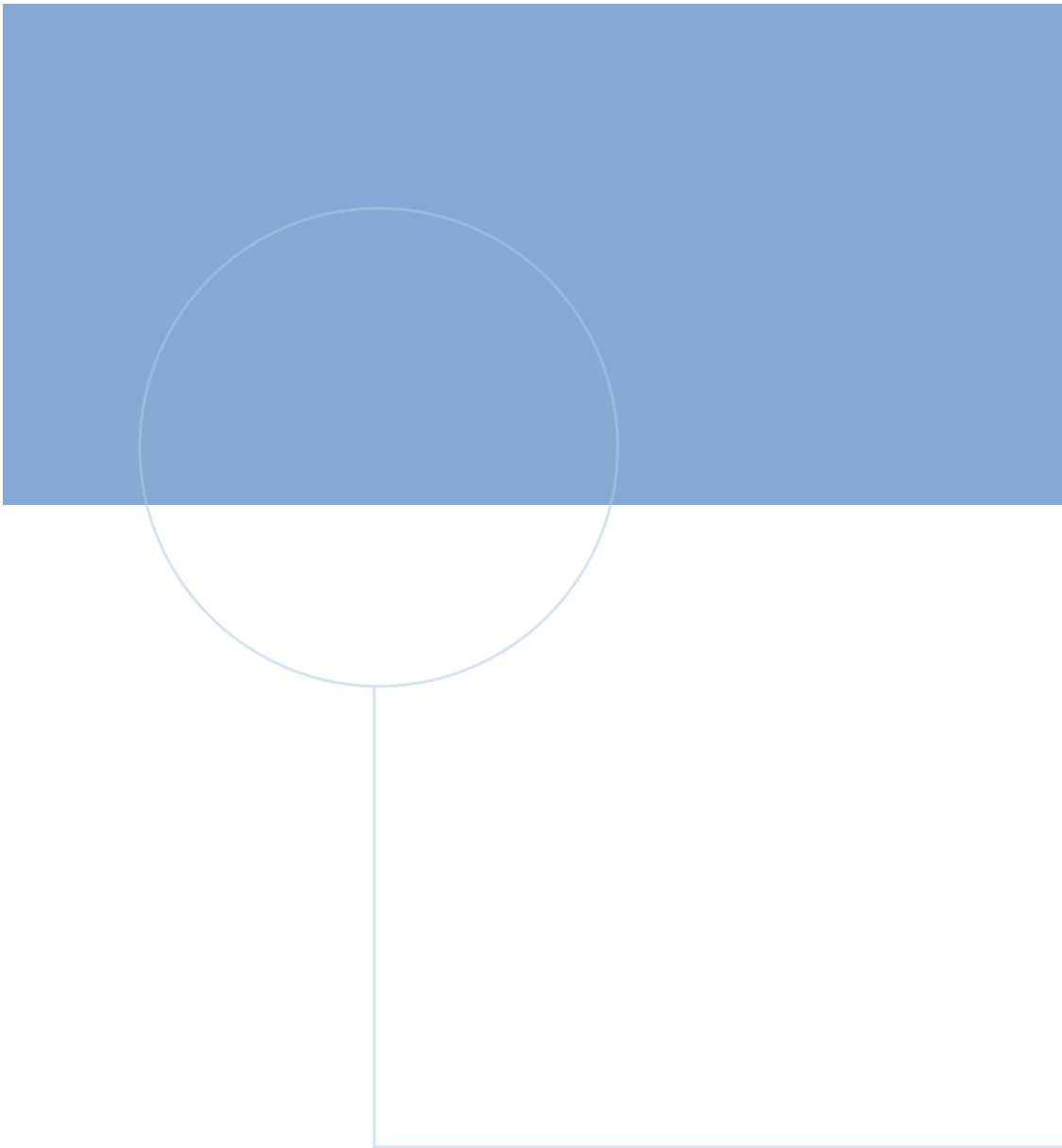
[20] D. Damljanovic, M. Agatonovic and H. Cunningham, 'Natural language interfaces to ontologies: Combining syntactic analysis and ontology-based lookup through the user interaction,' in *ESWC*, 2010.

[21] G. A. Miller, 'Wordnet: A lexical database for english,' *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[22] D. Diefenbach, T. P. Tanon, K. D. Singh and P. Maret, 'Question answering benchmarks for wikidata,' in *Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017.*, 2017. [Online]. Available: `http://ceur-ws.org/Vol-1963/paper555.pdf`.

[23] H. Bast and B. Buchhold, 'Qlever: A query engine for efficient sparql+ text search,' in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 647–656.

[24] A. Bordes, N. Usunier, S. Chopra and J. Weston, 'Large-scale simple question answering with memory networks,' *CoRR*, vol. abs/1506.02075, 2015. [Online]. Available: `http://arxiv.org/abs/1506.02075`.



NTNU

Norwegian University of
Science and Technology