Open Domain Question Answering over Hundreds of Linked Open Datasets

Eleftherios Dimitrakis

Thesis submitted in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science and Engineering

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Associate Prof. *Yannis Tzitzikas*

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

University of Crete
Computer Science Department

Open Domain Question Answering over Hundreds of Linked Open Datasets

Thesis submitted by
Eleftherios Dimitrakis
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author: _____
Eleftherios Dimitrakis

Committee approvals: _____
Yannis Tzitzikas
Associate Professor, University of Crete
Thesis Supervisor

_____
Dimitris Plexousakis
Professor, University of Crete
Committee Member

_____
Giorgos Flouris
Principal Researcher, FORTH-ICS
Committee Member

Departmental approval: _____
Antonios Argyros
Professor, Director of Graduate Studies

Heraklion, February 2019

Open Domain Question Answering over Hundreds of Linked Open Datasets

Abstract

Open domain Question Answering is a challenging task that requires, among others, to tackle the data distribution issue, i.e. the fact that datasets are scattered in several places. In this thesis, we focus on open domain Question Answering over Linked Open Data. We confine ourselves to three kinds of questions: factoid, confirmation, and definition questions. We introduce and comparatively evaluate information extraction based processes for question answering. The distinctive feature of our approach is that it can answer questions over millions of entities, by exploiting hundreds of Linked Data sources simultaneously, without having to use any training data. The process comprises three main phases: (i) Question Analysis (that includes question type identification and cleaning), (ii) Entities Detection, Named Entity recognition and linking and (iii) Answer Extraction (that includes RDF triples retrieval, scoring and matching). We demonstrate the benefits of this approach in terms of answerable questions and answer verification, and we investigate, through experimental results, how the steps of the question answering process affect the effectiveness of question answering. The evaluation was based on 1000 questions from SimpleQuestions and 2500 from QALD-7 Large-scale datasets.

Απάντηση Ερωτήσεων Ανοιχτού Πεδίου πάνω από εκατοντάδες Ανοιχτά Συνδεδεμένα Σύνολα Δεδομένων

Περίληψη

Η Απάντηση Ερωτήσεων Ανοιχτού Πεδίου (Open Domain Question Answering) αποτελεί πρόκληση, και απαιτεί μεταξύ άλλων, την αντιμετώπιση του προβλήματος της κατανομής των δεδομένων, δηλαδή του γεγονότος ότι τα σύνολα δεδομένων είναι διάσπαρτα σε πολλά διαφορετικά μέρη. Στην εργασία αυτή, εστιάζουμε στην Απάντηση Ερωτήσεων Ανοιχτού Πεδίου (Open Domain Question Answering), αξιοποιώντας Ανοιχτά Συνδεδεμένα Δεδομένα (Linked Open Data). Περιοριζόμαστε σε τρία (3) είδη ερωτήσεων: ερωτήσεις γεγονότων, ερωτήσεις επιβεβαίωσης και ερωτήσεις ορισμού. Παρουσιάζουμε και αξιολογούμε συγκριτικά, διαδικασίες που βασίζονται στην Εξαγωγή Πληροφοριών (Information Extraction) με σκοπό την απάντηση ερωτήσεων. Ιδιαίτερο χαρακτηριστικό της προσέγγισής μας είναι η δυνατότητα απάντησης επερωτήσεων για εκατομμύρια οντότητες, εκμεταλλευόμενοι ταυτόχρονα εκατοντάδες πηγές Συνδεδεμένων Δεδομένων (Linked Data), χωρίς να απαιτείται η χρήση δεδομένων για εκπαίδευση (training data). Η μέθοδος αποτελείται από τρεις κύριες φάσεις: (α) Ανάλυση Ερώτησης (η οποία περιλαμβάνει την αναγνώριση του τύπου της ερώτησης καθώς και τον καθαρισμό της), (β) Εντοπισμός Οντοτήτων, Αναγνώριση Ονομασίας Οντοτήτων και τη Σύνδεση αυτών με τις υποκείμενες πηγές, (γ) Εξαγωγή Απάντησης (η οποία περιλαμβάνει την ανάκτηση RDF τριπλετών, τη βαθμολόγηση αυτών και την εξαγωγή της καλύτερης τριπλέτας). Επιδεικνύουμε τα οφέλη της προσέγγισης αυτής όσον αφορά το πλήθος των ερωτήσεων που μπορούν να απαντηθούν καθώς και την επαλήθευση των απαντήσεων. Επίσης ερευνούμε, μέσω πειραματικών αποτελεσμάτων, πως επηρεάζουν τα βήματα της Απάντησης Ερωτήσεων (Question Answering) την αποτελεσματικότητα. Η αξιολόγηση βασίστηκε σε 1000 ερωτήσεις από το SimpleQuestions σύνολο δεδομένων καθώς και 2500 ερωτήσεις από το QALD-7 Large-Scale σύνολο δεδομένων.

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον επόπτη καθηγητή μου κ. Γιάννη Τζίτζικα για την ορθή καθοδήγηση και ουσιαστική συμβολή του στην ολοκλήρωση της παρούσας μεταπτυχιακής εργασίας. Ακόμη θέλω να εκφράσω τις ευχαριστίες μου στον κ. Δημήτρη Πλεξουσάκη και στον κ. Γιώργο Φλουρή για την προθυμία τους να συμμετέχουν στην τριμελή επιτροπή. Ακόμα ευχαριστώ το Ινστιτούτο Πληροφορικής του Ιδρύματος Τεχνολογίας και Έρευνας για την πολύτιμη υποστήριξη σε υλικοτεχνική υποδομή και τεχνογνωσία, καθώς και για την υποτροφία που μου προσέφερε κατα τη διάρκεια της μεταπτυχιακής μου εργασίας. Στο σημείο αυτό θα ήθελα να ευχαριστήσω τους γονείς μου και την οικογένειά μου για την συμπαράσταση και την υποστήριξη που μου έδωσαν όλα αυτά τα χρόνια. Επειτα θα ήθελα να ευχαριστήσω τους συναργάτες και φίλους από το Ινστιτούτο Πληροφορικής του Ιδρύματος Τεχνολογίας και Έρευνας που ήταν πάντα πρόθυμοι να με βοηθήσουν καθ΄ όλη την διάρκεια των σπουδών μου.

στους γονείς μου

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

Question Answering systems, aim at supplying precise answers to user questions posed in Natural Language. They are appropriate, in cases where the user seeks for concise and specific answers. Furthermore, such systems can either be *closed-domain* i.e. focus on answering questions under a specific domain or *open-domain* i.e. answering questions about "anything". Overtime, the different types of knowledge source exploited for the search and extraction of these answers include: (a) structured data i.e. RDF graphs, SQL databases etc., (b) unstructured data i.e. collection of documents, (c) hybrid i.e. combination of structured and unstructured data.

Open domain Question Answering (QA) is a challenging task because it requires tackling (i) the data distribution issue (since several datasets have to be considered for supporting open domain question answering), (ii) the increased difficulty of word sense disambiguation (since the associated vocabulary is not restricted to a specific domain), and (iii) the difficulty (or inability) to apply complex techniques (e.g. deep NLP analysis to the available resources) due to the huge size of the data that have to be considered.

In this thesis, we focus on Open Domain Question Answering over *Linked Data*. Since there are hundreds of datasets published as Linked Data in various domains, we definitely need a method for tackling the distribution issue. To this end, we propose an approach that exploits LODsyndesis [36], which is a recently launched suite of services over hundreds of LOD Datasets, millions of entities, and billions of facts. This choice is of primary importance, since it can bring benefits to QA. Specifically, (a) it allows verifying an answer to a given question from multiple sources, and (b) it increases the number of questions that can be answered, since many datasets contain complementary information for the same entities.

Regarding (a), consider the question *"What is the population of Kyoto?"*. The system retrieves two candidate triples {(Kyoto, population, 1,474,570), (Kyoto, population, 1,500,000)} containing different values for the same relation, however, it returns as answer the first triple, since it has two provenance datasets (Geonames and NYtimes) instead of the second triple which has one provenance dataset

(DBpedia).

Regarding (b), consider the questions *"Did Aristotle influence Ioannes Georgius Gadamer?"* and *"Did Aristotle influenced Al Farabi?"*. Both questions can be categorized to the same topic, i.e. *people influenced by Aristotle*, however there is no single knowledge source that contains the essential information for answering both questions. Specifically, the first question can be answered by a triple which can be found in both DBpedia and YAGO knowledge bases, while the second one can be answered by a triple which occurs only in Freebase knowledge base.

In comparison to other LOD-based QA approaches, apart from the aforementioned distinctive characteristics, in our approach we do not rely on any training data.

We introduce and evaluate a process for answering a natural language question, that comprises three main phases: *(i) Query Analysis*, *(ii) Entities Detection* and *(iii) Answer Extraction*. The first phase includes the tasks of question cleaning and analysis, as well as the identification of the *question type*. The second phase includes the tasks of the recognition of the question *entities* and their linking with their corresponding URIs in the underlying sources, by exploiting the LODsyndesis services and the Named Entity recognition and linking capabilities of widely used tools (specifically Stanford CoreNLP [22, 30] and DBpedia Spotlight [31]). The first tool, for short SCNPL is based on a combination of hand-crafted rules and statistical sequence taggers for recognizing named entities. The second one, is based on a string matching algorithm for spotting entities, a lexicalization dataset for retrieving candidates and a variation of TF*IDF for disambiguating the final entities and their URIs. The third phase, includes the retrieval of candidate RDF triples and the extraction of the best matching triple, for producing the answer, based on the contents of various datasets (from LODsyndesis).

As regards the *Entities Detection* step, since we exploit two different tools for the Named Entity Recognition task, one important question is to understand how the way the capabilities of these tools are used, affects the outcome of the process. To combine the SCNLP and DBpedia Spotlight in an "optimal" way, we elaborate on this issue and we report comparative results. As regards the *Answer Extraction* step, our aim is to tackle lexical gap between the question and the underlying sources by expanding the available set of question words. The expansion is achieved by exploiting the lemmas (from SCNLP) of the question words, and then, based on the POS tag of each word, if (a) it is a Verb, we retrieve all the derived nouns (from WordNet), (b) if it is a Noun, we retrieve all the derived verbs. To this end, we elaborate on this issue and we report comparative results for understanding how each expansion step affects the overall approach. The evaluation results so far shows that our approach is KB agnostic in the sense that it is applicable in any given KB (index by LODsyndesis) without any additional effort, achieving similar results with approaches requiring training data.

In a nutshell, the key contributions of this thesis are: (a) we describe an approach for open domain QA that exploits the wealth of data coming from hundreds of datasets (and does not depend on the availability of training data), (b)

we demonstrate the benefits of this approach in terms of answerable questions and answer verification, (c) we investigate how the steps of the QA process affect the effectiveness of QA, and (d) we report experimental results over 1000 questions from SimpleQuestions and 2500 from QALD-7 Large-scale datasets. The results show, (i) the importance of considering the "lexical gap" between the input question and the underlying sources, for the retrieval of relevant information, and (ii) the importance of using both large-scale KB-based tools (i.e. DBpedia Spotlight) and KB-agnostic tools (i.e. SCNLP) for the Named Entity recognition and linking, for achieving competitive results for open domain question answering.

1.1 Outline of Thesis

The rest of this thesis is organized as follows: Chapter 2 discusses the context and the background, and describes related work and the placement of our approach in the landscape. Chapter 3 describes the proposed approach and details (i) the Knowledge services used, (ii) the QA process followed and (iii) the elaboration on problematic tasks of the QA process. Chapter 4 describes the evaluation datasets and metrics used and reports the evaluation results. Chapter 5 describes the available way for exploiting the current work. Finally, Chapter 6 concludes the thesis and identifies directions for future research.

Chapter 2

Context and Related Work

2.1 Context and Background

2.1.1 Context: Faceted Search (FS)

It is an interaction framework based on a multi-dimensional classification of data objects and it is the de-facto standard in e-commerce (e.g. eBay, booking.com) and tourism services. It allows users to browse and explore the information space in a guided, yet unconstrained way through a visual interface [43]. The key features of this framework include: (a) display of the current results in multiple categorization schemes (called facets, or dimensions, or just attributes), (b) display of only the facets and values leading to non-empty results, (c) display of the count information for each value (i.e. the number of results the user will obtain by selecting each value), and (d) ability of gradual focus refinement, i.e. it is a session-based interaction paradigm in contrast to the stateless query-and-response dialogue of most search systems. Faceted search has been proposed and applied for web searching, (e.g. [37]), for semantically enriching web search results, (e.g. [17]), for patent-search, (e.g. [18]), as well as for exploring RDF and Linked Data (e.g. see [20, 46], as well as [49] for a recent survey).

The enrichment of faceted search with *preferences*, hereafter *Preference-enriched Faceted Search* (PFS), was proposed in [50, 38]. It offers actions to the user enabling the ordering of the facets, values, and objects, using *best, worst, prefer to* actions (i.e. relative preferences), *around to* actions (over a specific value), actions that order them lexicographically, or based on their values or even count values. Furthermore, the user is able to *compose* object related preference actions, using *Priority, Pareto, Pareto Optimal* (i.e. skyline) and other. The distinctive features of PFS is that it allows expressing preferences over attributes, whose values can be hierarchically organized (and/or multi-valued), it supports preference inheritance, and it offers scope-based rules for resolving automatically the conflicts that may arise. As a result the user is able to restrict his current focus by using the faceted interaction scheme (hard restrictions) that lead to non-empty results, and rank the objects of his focus according to the expressed preferences. Recently, PFS has been

used in various domains, e.g. for offering a flexible process for the identification of fish species [48], as a Voting Advice Application [47], as well as, for data that contain also geographical information [26].

2.1.2 Context: Conversational Faceted Search

It is an extension of Faceted Search with an attached speech interface on top of the paradigm. Only a few works exist, for example [13], exploits a speech interface over facets that index audio metadata associated with audio content. This system is used for the Spoken Web and the associated Medieval Spoken Web Search Task [32]. A faceted browser over datasets, available in the Linked Open Data (LOD) cloud is described in [28]. The user interacts with the system through voice commands that are then translated to SPARQL queries using NLP and submitted to LOD end-points.

To the best of our knowledge though, the only work that combines spoken dialogue systems with faceted search is the one presented in [39], where the described LD-SDS system is limited to spoken dialogues over structured datasets (expressed in RDF), while [16] studied the case where the user comments or reviews that are associated with the items, can be exploited in the context of the dialogue (i.e. for answering those user queries that cannot be answered by the "core" dataset).

2.1.3 Background: LODsyndesis

LODsyndesis is a suite of services and tools, over hundreds of LOD Datasets, millions of entities, and billions of facts, that helps the user to exploit the Linked Open Data (LOD) cloud. Its key characteristics is that it has indexed the content of all datasets in the LOD [35, 36] and its knowledge graph contains all the inferred equivalence relationships, which occur between entities and schemas. The provided services aid the following tasks: a) Dataset discovery i.e. it enables content-based dataset discovery based on a specific dataset (e.g. "find the K datasets that are more connected to dataset X"). b) Object/Entity co-reference i.e. retrieve complete information and their provenance, about a specific entity (or even a set of entities) identified by a URI. c) Data quality and veracity assessment, i.e. assessing the connectivity between any set of datasets, and estimating data veracity, by exploiting the cross-dataset inference, that allows spotting the contradictions that exist among different datasets. d) dataset enrichment and/or visualizations. e) other tasks.

2.2 Related Work

Here, at first we introduce and discuss the Question Answering systems in general, in §2.2.1, while §2.2.2 focuses on related works in the area of Question Answering systems over Knowledge Bases. Finally, §2.2.3 presents the placement of the thesis in the landscape.

2.2.1 Question Answering (QA) Systems

Question Answering (QA) systems, aim at supplying precise answers to user questions posed in Natural Language (NL). They are appropriate, in cases where the user seeks for concise and specific answers [33, 2]. Figure 2.1 shows a categorization of such systems based on four aspects.

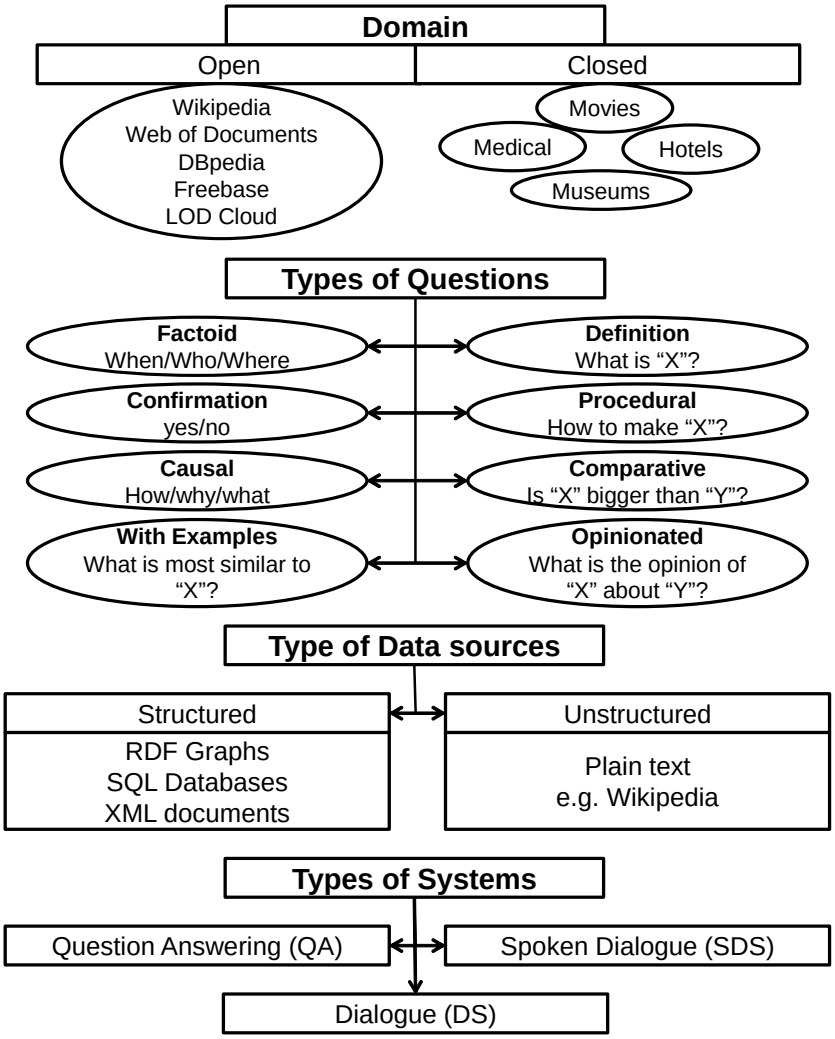

Figure 2.1: **Aspects of the Landscape:** (1) Domain of Knowledge (2) Type of Question (3) Type of Knowledge Source (4) Type of System

(a) Domain: Question Answering (QA) systems can be either *closed-domain* i.e. focus on answering questions under a specific domain Medical, Culture etc. [12], [27]. Or *open-domain* i.e. systems which do not focus on a specific search area, but in contrast, their purpose is to answer questions about "anything" [21], [53], [10], [33]. The techniques that could be used for the QA implementation

between those two types differ in many aspects. As stated in [34] the aspects are three: (i) size of available data, (2) context of the available knowledge, (3) kind of resources to be exploited i.e. domain-specific vocabularies etc.

(b) Types of Questions: There is a wide range of question types supported by QA systems, requiring different approaches for being able to answer them. As it is mentioned in [33], we can categorize the types of natural language questions as follows:

1. *Factoid (when/who/where):* questions that essentially require a single fact or a small piece of text to be returned as answer.
   - e.g. "When did the WWI begun?"
2. *Confirmation (yes/no):* questions that require a yes/no answer.
   - e.g. "Is Athens the capital of Greece?"
3. *Definition:* questions that require answers that are definitions of terms.
   - e.g. "What is the Resource Description Framework?"
4. *Causal (how/why/what):* questions that require that the answer should be one or more consequences of a fact.
   - e.g. "What are the consequences of Iraq War?"
5. *Procedural:* questions that require a set of actions needed for accomplishing something.
   - e.g. "Which are the steps to get a master degree?"
6. *Comparative:* questions that require as an answer, a set of differences between two or more subjects.
   - e.g. "Which are the differences between SSD and HDD?"
7. *With Examples:* questions that target to find examples that best describe the reference point of the question.
   - e.g. "Which are the most similar disks to hard disk X?"
8. *Opinionated:* questions that require to find the opinion of someone about a subject or a fact.
   - e.g. "Which is the opinion of the Americans about the Iraq war?"

(c) Type of Data Source: The different types of data sources exploited by QA systems for the extraction of relevant information based on a question can be categorized in three main types: *(1) Documents* [9, 53, 52, 54], *(2) Data*

[29, 3, 24, 62, 6, 56, 61] and *(3) Hybrid* (data and documents) [40, 11, 19, 55, 44]. The approaches followed in each case for the analysis of the information, differ a lot due to the nature of each type. For example in the Data source case, we do not need to apply complex NLP techniques, since their structure is appropriate to capture semantic specifications of the information stored (e.g. Albert Einstein is a Person). In contrast, the Documents case, requires deep NLP and NLU techniques for extracting such kind of information. As regards the Hybrid knowledge representation, it requires more effort, since both structured and unstructured knowledge needs to be used, with and additional step to combine the information and provide a unique global answer.

(d) Type of interaction: Based on the interaction type offered by the system we can categorize it between: (i) QA system i.e. the user provides a single query and receives a single precise answer, without enabling a multi-step interaction. (ii) Dialogue QA system, where the user has the ability to interact with the system in a multi-step procedure in order to refine and fulfill his/her information needs. (iii) Spoken Dialogue QA system, where is an extension of the Dialogue type with an attached spoken interface on top of the system.

2.2.2 Question Answering over Knowledge Bases

Over time, Knowledge Base Question Answering (KBQA) systems have converged to two major approaches: (a) *Semantic Parsing (SP)* [4, 5, 60, 6, 42, 59, 23], and (b) *Information Extraction (IE)* [57, 1, 3, 41, 15].

On the one hand, the *SP* approaches focus on question understanding and therefore attempt to convert sentences into their semantic representation, such as logical forms. They are capable of answering compositional questions, since they are able to represent and compose aggregation operators e.g., *argmax*, *count* etc., however, they suffer in cases where the KB structure of a statement differs from the one that the semantic parser formed.

On the other hand, *IE* approaches aim at identifying topic/focus entities in the input question and then, via either pre-defined or automatically generated templates, map the question to the KB predicates. Finally, they explore the neighbourhood (in the graph of the Knowledge Base) of the matched entities to extract the answer. Note that *IE* approaches cannot answer compositional questions, since they cannot represent the respective operators [14, 25]. As stated in [44], such systems, rely on a lexicon, learned from labeled training data, or even supported by additional resources, such as question paraphrases [5] and weakly labeled sentences from a large text collection [58]. However, the size of such training data tends to be very small compared to the number of distinct predicates, literals etc. in the KB and consequently the produced lexicons have limited coverage. Our work falls in the IE category, however, we do not need to use any training data.

The approaches that are more related to our work presented in this thesis, are *WDAqua* [15], *AMAL* [41], both participants of the QALD-7 challenge. The choice was made based on a few key-points that make those systems competitive

approaches to our approach. First, we concerned about systems that require no training data and second, systems that can answer Simple Questions (termed in this way in [7]), i.e questions that relate to an entity and a property of this entity.

As regards the first key-point (no training data), we chose *WDAqua* [15], which is a multilingual QA system that supports different RDF KBs like DBpedia and Wikidata and understands both natural language and keyword queries. It requires no training data, as mentioned in [51] (except of step (4) which is optional). Query answering is achieved through the following steps: (1) Query Expansion, that aims to identify to which possible concepts in the KB the question words can refer to. (2) SPARQL Query Generation, using a combinatorial rule-based approach given an input question, by exploiting the semantics of the supported KBs, (3) SPARQL Query Ranking, using features like the number of words in the input question that are associated with a SPARQL query resource and how similar those two are, and (4) Answer Decision, which needs a few training data and uses a model that predicts, based on the previous features, whether the generated query is an answer or not.

As regards, the second key-point (Simple Questions), we chose *AMAL* [41], which is a French QA system over DBpedia. It receives input queries in French, which are analyzed and answered with information found in DBpedia. It mainly focuses on answering Simple Questions. It follows the steps of: (1) Question type identification via pattern matching (it supports boolean, date, number and resource types of questions, additionally it supports list and aggregation questions for some of the above types), (2) Entity Extraction using syntactic parsing and Entity linking to DBpedia, and (3) Property extraction by removing the found entity and then searching matches in DBpedia properties with the remaining words. For this last task they also exploit Wikipedia Disambiguation links if possible and a manually crafted lexicon of DBpedia properties, linked to one or more possible French expressions.

Another noteworthy related work, that is not from the aforementioned challenge, is the system *Aqqu* [3], based on Freebase, which is an *Open-domain* factoid KBQA system. It follows the steps of: (1) Entity Identification (retrieval of candidate entities), (2) Template Matching (construction of candidate queries based on three handcrafted templates), (3) Relation Matching (finding of candidate predicates of the KB that match the input question's phrases), and finally (4) Rank Queries (finding the most suitable query based on a set of features). The system evaluation over the evaluation dataset Free917 achieves accuracy of 0.66-0.76. In the collection WebQuestions it achieves average F1 score of 0.5.

Another approach, [61], which is an *Open-domain* KBQA system over a few KBs (e.g. SIDER, Diseasome and Drugbank), proposes a joint method based on Integer Linear Programming (ILP). First the candidate entities are retrieved by a simple string matching algorithm, i.e. Levenshtein distance. This method jointly considers the two interactive tasks of the Alignment Construction (between the candidate entities of the aforementioned KBs) and the Query Construction. Thus,

the process of aligning the candidate entities is applied at query time, simultaneously with the disambiguation step, which is solved by an ILP program.

A more recent approach, SINA [45], which is a QA system over a few interlinked datasets (e.g. SIDER, Diseasome and Drugbank), falls in the IE category and requires training data to be tuned. It follows the steps of: (1) Query preprocessing, by applying to the input question, tokenization, stop-word removal and lemmatization of the individual tokens. (2) Segment validation, for grouping tokens, in order to capture multi-word expressions. (3) Resource retrieval, by string matching over the rdf:label of the recources. Here a lightweight reasoning for inferring owl:sameAs relationships is also applied. (4) Disambiguation, where the best subset of resources, between the candidates, are selected. (5) Query Construction, to construct a SPARQL query. (6) Representation, which presents the retrieved results after the query evaluation.

In contrast to the above works, WDAqua [15], AMAL [41], Aqqu [3], [61], and SINA [45], where a single or few KBs are supported, our system provides access to 400 datasets enriched with inferred equivalence relationships.

As a final remark we should note that only a few of the aforementioned works, provide efficiency results for their approaches.

2.2.3 Our Placement in the Landscape

The distinctive feature of our approach (i.e. LODQA) is that it supports open domain LOD-based Question Answering for three kinds of questions (factoid, confrmation, and defnition) with no training data, over hundreds of datasets.

In contrast to the most related works (WDAqua [15], AMAL [41], and Aqqu [3]) where a single or few KBs are supported, LODQA provides access to 400 datasets enriched with inferred equivalence relationships. Indeed, LODQA can answer three kinds of questions (factoid, confirmation, and definition) over millions of entities, by exploiting these 400 datasets containing in total billions of facts, simultaneously, without requiring any training data. Moreover, LODQA has the ability to verify the answers from several datasets. Furthermore, the large size of the underlying knowledge exploited by our system, make infeasible the support of SPARQL queries for retrieving relevant information. Therefore, it differs from all the aforementioned approaches in this step, in the sense that we use an index not SPARQL querying.

In comparison to WDAqua [15], LODQA takes into account the syntactic form of the question and the relations of the question words instead of solely relying on the semantics of the question words. Also, LODQA does not support multilinguality and keyword queries.

In comparison to AMAL [41], LODQA uses the LODsyndesis service that provides equivalent relationships and WordNet synonyms, instead of Wikipedia Disambiguation links and the DBpedia lexicons for relation matching that is used by AMAL. In this way LODQA exploits multiple sources for this task, instead of relying solely on DBpedia resources. However, LODQA supports less question

types, since it does not support list and aggregation questions and it does not distinguishes the factoid questions in more fine grained types (e.g. resource, date and number).

In comparison to Aqqu [3], LODQA exploits both DBpedia Spotlight and Stanford CoreNLP, for the entity identification, instead of relying solely to hand-crafted rules based on POS-tags. Moreover LODQA does not use any training data or hand-crafted features for the extraction of the final answer.

In comparison to [61] and SINA [45], which perform the interlinking of resources between the KBs at query time, however only for a few datasets, in our approach the interlinking has been done only once, at indexing time, and involves 400 datasets.

Chapter 3

Approach

3.1 The Proposed Approach

In this section, we describe the proposed approach. In particular, in §3.1.1, we present the LODsyndesis services that we exploit, while in §3.1.2, we introduce the proposed QA process. In §3.1.3 we present the elaboration on problematic tasks of the QA process. Finally, Figure 3.1 shows the running example.

3.1.1 Knowledge Services

We decided to use *LODsyndesis*, for the two tasks, namely *Entity Detection* and *Answer Extraction*, due to the following benefits that cannot be found in a single knowledge base: (a) it collects all the available information for millions of entities from hundreds of datasets, (b) it contains complementary information from different datasets, and (c) it can surpass the problems of non-informative URIs. The process of global indexing of LODsyndesis is shown in Figure 3.2, i.e., LODsyndesis uses as input several datasets containing RDF triples, where a triple is a statement of the form subject-predicate-object (s,p,o) and T is the set of all the triples in universe. Moreover, it uses several equivalence relationships (e.g., `owl:sameAs` relationships denote that two URIs refer to the same entity), and it computes their transitive and symmetric closure for collecting all the information for an entity (e.g., see the index for "Kyoto" in Figure 3.2). Concerning benefit a), it is important for any kind of question to verify the answer from several sources. Regarding benefit b), for any type of question, two or more datasets can possibly answer different questions, e.g., in Figure 3.2, one dataset contains a description about Kyoto, another one about a Kyoto Museum, etc. Therefore, if we use only one of these datasets, we will not be able to answer both questions. Concerning benefit c), many datasets publish non-informative URIs, e.g., suppose that a user asks a question "Is Nintendo located in Kyoto?". In Figure 3.2, only Wikidata contains that information, and the corresponding triple is the following: (wikidata:Q34600,wikidata:P276, wikidata:Q8093). However, LODsyndesis stores the equivalent URIs of each URI, thereby, it knows that dbp:Kyoto `owl:sameAs`

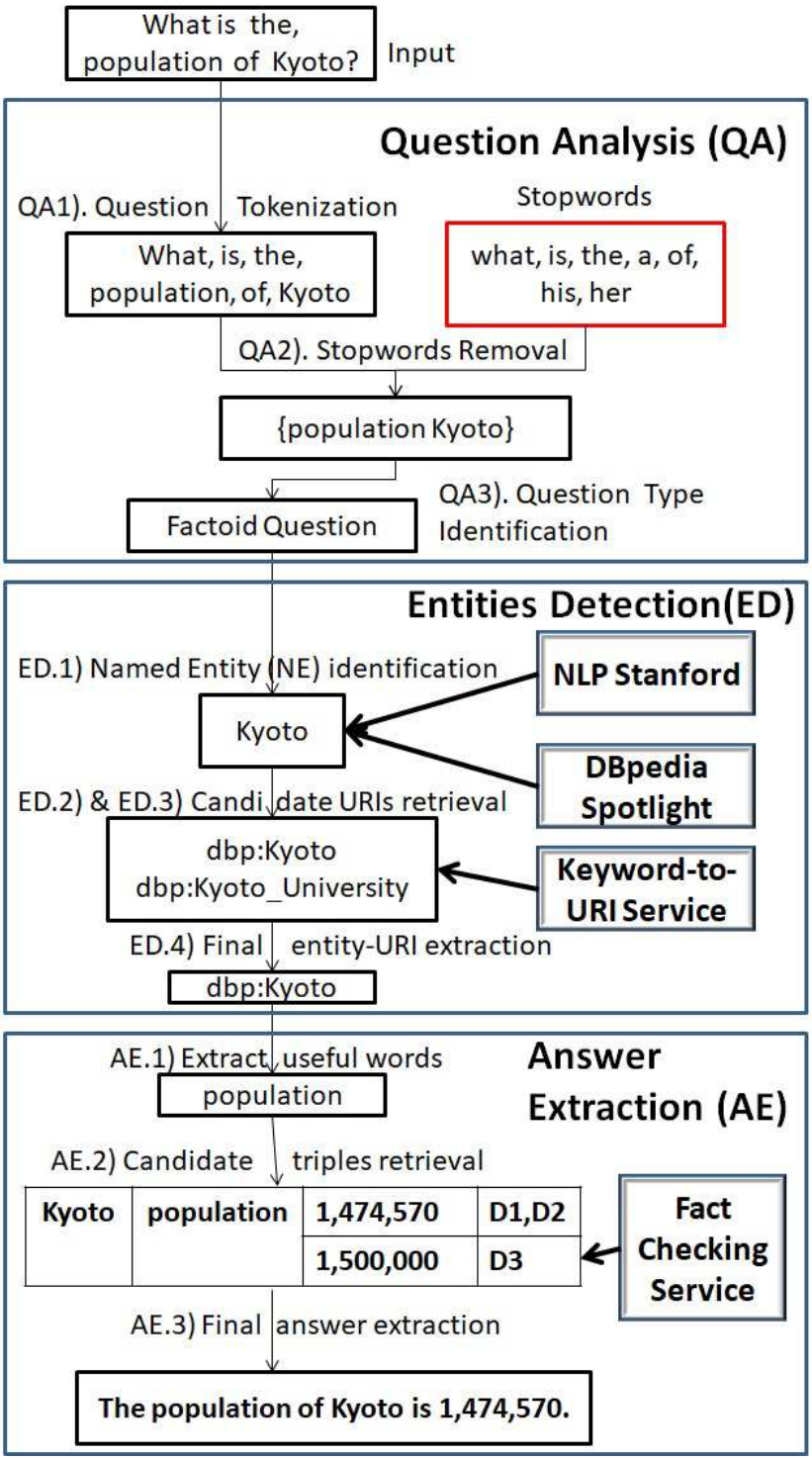

Figure 3.1: The running example for answering a specific query through our approach

wdt:Q34600, dbp:isLocatedIn `owl:sameAs` wdt:P276 and wdt:Q8093 `owl:sameAs` test:Nintendo. Therefore, we can find fast the correct answer, by checking the equivalent URIs of each one.

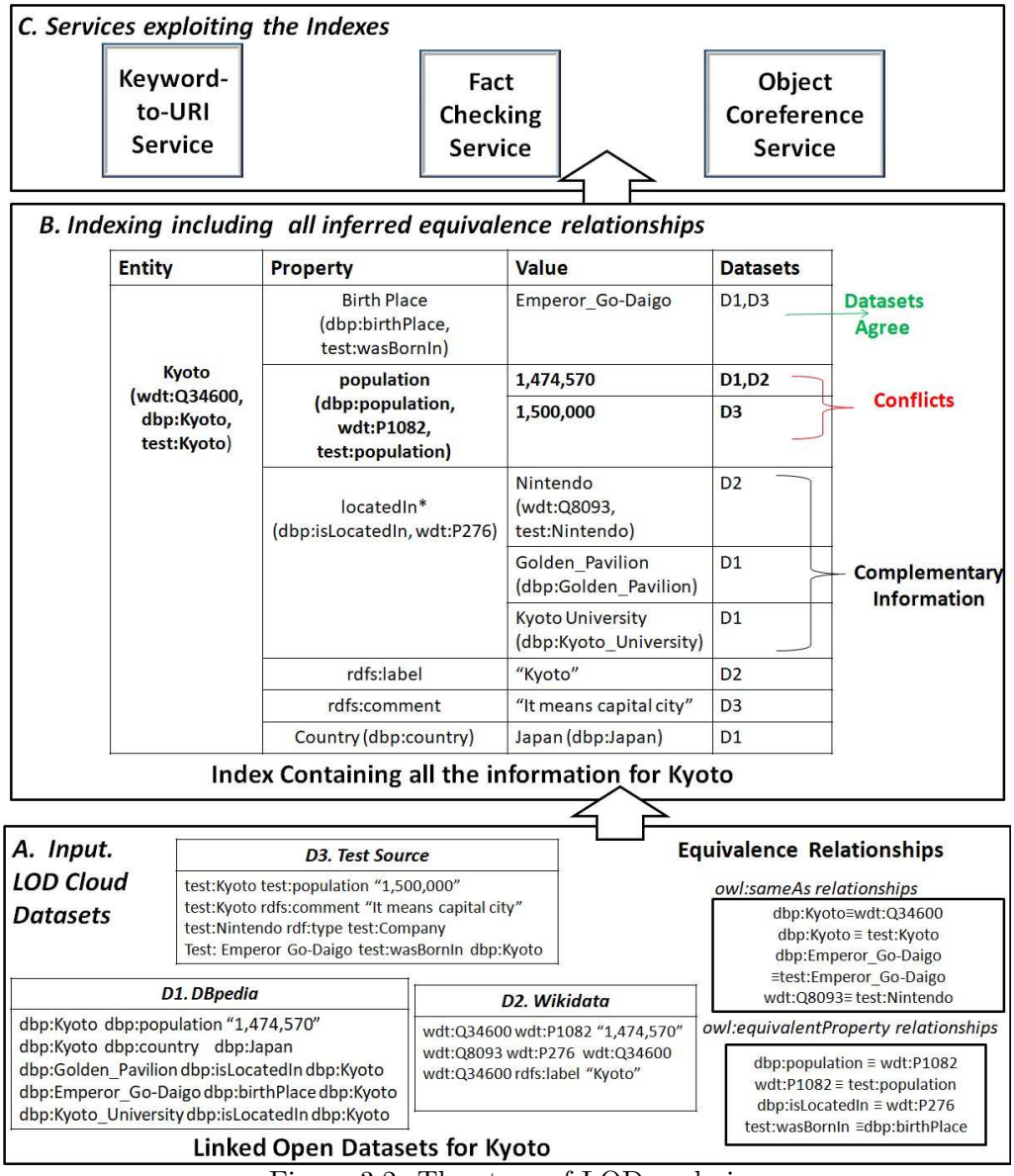

Figure 3.2: The steps of LODsyndesis

3.1.2 The QA Process

Let q be a user's question, which is not answerable by the core dataset. Therefore, we exploit external LOD, specifically, LODsyndesis. The process contains three main phases,

Query Analysis (QA), *Entities Detection (ED)* and *Answer Extraction (AE)*.

Figure 3.3 shows an overview of the process. Each of the above phases are described below, along with a corresponding algorithm (see Alg. 1, 2 and 3 respectively).

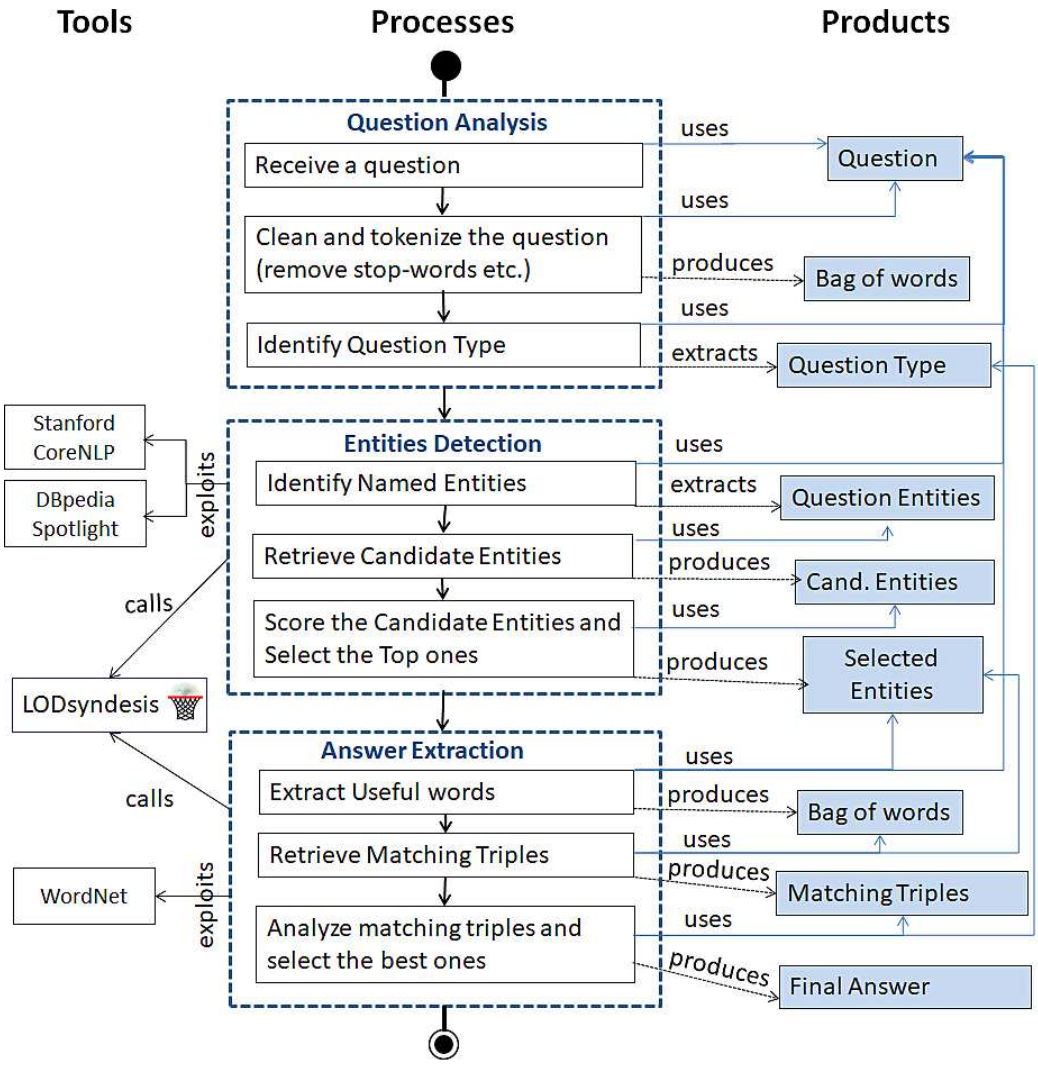

Figure 3.3: Overview of the QA Process

Step **QA.1)** ***Question tokenization***. Here we split the question to a set of individual words, denoted by W_q (where $W_q \subset W$ the set of all possible words). In our example in Fig. 3.1, W_q = {What, is, the, population, of, kyoto, ?} (see line 1 in Alg. 1).

Step **QA.2)** ***Stopwords removal***. Here we remove the set of stopwords, $StopW$,

and define $W_q^u = W_q \setminus StopW$. In our example in Fig. 3.1, W_q^u = {population, kyoto} (see line 2 in Alg. 1).

Step **QA.3)** ***Question type identification (factoid, confirmation, definition)***. Here we identify the question type through indicative words and simple heuristics. For *factoid* questions, we use a set of starting words denoted as $W_{factoid}$ = {when, who, where, what, which, ...}, while for the *confirmation* questions, the set of starting words used is $W_{confirm}$ = {are, did, is, was, does, were, do, ...}. For *definition* questions, we use two distinct sets, (i) a set of starting words denoted by $W_{defStart}$ = {what is} and (ii) a set of words that should be contained in the question, denoted as $W_{defCont}$ = {mean, meaning, definition} (see line 3 in Alg. 1). In our example (see Fig. 3.1), the identified question type is *factoid*, i.e., it starts with the word "what".

Procedure 1 The process of the *Question Analysis* module

Input: A query q, and a set of stopwords $StopW$
Output: A question analysis object qa_obj with attributes q, W_q^u, $type_q$
1: $W_q \leftarrow split(q)$
2: $W_q^u \leftarrow W_q \setminus StopW$
3: $type_q \leftarrow identifyQuestionType(q, W_q^u)$
4: $qa\_obj \leftarrow QuestionAnalysis(q, W_q^u, type_q)$
5: **return** qa_obj

Step **ED.1)** ***Named Entity (NE) identification in the input question***. Here, we leverage a Named Entity Recognizer (NER) and a Named Entity Linker (NEL) to identify and extract Named Entities in the input question. For NER we use SCNPL (Stanford coreNLP) [22, 30], while for NEL we use DBpedia Spotlight [31]. As regards the former, it is used for detecting Named and Numerical Entities in plain text (e.g. that Kyoto is a *Location*, 11-01-2018 is a *Date*, etc.). It exploits a combination of three Conditional Random Fields sequence taggers trained on various corpora annotated with NEs, to identify NEs, whereas Numerical Entities are recognized using a rule-based system. As regards, DBpedia Spotlight, it annotates free text with identified DBpedia URIs. It uses an extended set of labels and a string matching algorithm with longest case-insensitive match to improve Entity spotting, extracts candidates for each spotted Entity by leveraging DBpedia Lexicalization dataset and finally disambiguates between the candidates by scoring them using a proposed $TF*ICF$ (Term Frequency - Inverse Candidate Frequency) formula. We use SCNLP to extract each identified NE word $w \in W_q^u$, and to construct a set of entities E_q^s. We use DBpedia Spotlight to extract each identified NE word, i.e., $w \in W_q^u$, along with its corresponding URI in DBpedia denoted as u. Then, we construct a set of pairs (entity,URI) i.e., EU_q^d. In our example of Fig. 3.1, (a) E_q^s = {Kyoto} and (b) EU_q^d = { (Kyoto, http://dbpedia.org/resource/Kyoto) } (see lines 1-2 in Alg. 2).

Step **ED.2)** ***Candidate URIS retrieval for the question entities identified by SCNLP***. As it is described in lines 3-6 of Alg. 2, for each NE of the question identified by SCNLP, i.e., $e \in E_q^s$, we retrieve a set of candidate URIs denoted as $U(e)$, by calling *keyword_to_URIs* service from *LODsyndesis* (e.g. $U(Kyoto) = \{dbp : Kyoto, dbp : Kyoto\_University\}$). Then, we construct a set of pairs $(e, U(e))$, denoted as cEU_q^s (e.g. cEU_q^s = { (Kyoto,{dbp:Kyoto, dbp:Kyoto_University}) }).

Step **ED.3)** ***URI matching for the question entities identified by SCNLP*** Here, for each NE of the question identified by SCNLP, i.e., $e \in E_q^s$ we extract the best matching URI among the retrieved candidates, i.e., $U(e)$, using the minimum Levenshtein distance (e.g., the best match for the entity "Kyoto" is "dbp : Kyoto"). Then, we construct a set of pairs $(e, minLevURI(e))$, denoted as EU_q^s, e.g. EU_q^s = { (Kyoto, dbp:Kyoto) }, (see lines 7-11 in Alg. 2).

Step **ED.4)** ***Final entity-uri extraction based on DBpedia Spotlight and SCNLP*** Here, we exploit the two sets of entities and their corresponding URI extracted by: (a) SCNLP i.e., EU_q^s and (b) DBpedia Spotlight i.e., EU_q^d. We construct a set of pairs (entity, final entity URI) denoted as EU_q^c as follows. For each entity identified by both tools (i.e. the common entities of the two sets) we extract the best identified URI using the maximum Jaccard similarity between the URI suffix (i.e., the last part of the URI) and the question, leading to a pair (entity,maxJaccard(URI)). For the rest of the entities (i.e. identified either by SCNLP or DBpedia Spotlight) we extract the pair (entity,matched(URI)). The process is described in lines 12-22 of Alg. 2.

Step **AE.1)** ***Extract useful words words for the triple retrieval***. Here, we first remove from the set of useful words (i.e., W_q^u) the identified question entities i.e., $E_q^c = \{e \mid (e, u) \in EU_q^c\}$, for constructing a set of words denoted as W_q^o, i.e., $W_q^o = W_q^u \setminus E_q^c$. The next step, for tackling the lexical gap between the input question and the underlying sources, is to expand the set of words W_q^o. First, we include the lemma of each word in this set, using the SCNLP lemmatizer [22, 30]. Then, for each word in the expanded W_q^o which is: (a) a verb, or (b) a noun, we retrieve all derived nouns (e.g. produce-producer) and verbs (e.g. publisher-publish) respectively, from WordNet dictionary and include them in the set of words W_q^o. In our example, W_q^o = {population}, as it can be seen in Alg. 3 (lines 1-4).

Step **AE.2)** ***Candidate triples retrieval***. Here we select a single entity from E_q^c, since the *factChecking* service of LODsyndesis takes as input a single entity, together with a set of words for checking a fact, i.e., it is a function $fct : (U, W) \rightarrow T$. Indeed, we select the URI of a selected entity e_{sel}, say u_{sel}, where $u_{sel} \in E_q^c$, and we put the URIs for the rest of the entities in W_q^o set, i.e., $W_q^o = W_q^o \cup \{\cup_{e \in \{E_q^c \setminus u_{sel}\}} u_e\}$.

Procedure 2 The process of the *Entities Detection* module

Input: A question analysis object qa_obj

Output: A set of pairs (entity, URI) EU_q^c

1: $E_q^s \leftarrow StanfordCoreNlp.ner(qa\_obj.q)$
2: $EU_q^d \leftarrow DBpediaSpotlight.annotate(qa\_obj.q)$
3: $cEU_q^s \leftarrow \emptyset$
4: **for all** $e \in E_q^s$ **do**
5: $\quad U(e) \leftarrow keyword\_to\_URIs(e)$
6: $\quad cEU_q^s \leftarrow cEU_q^s \cup \{(e, U(e))\}$
7: $EU_q^s \leftarrow \emptyset$
8: **for all** $e \in E_q^s$ **do**
9: $\quad U(e) \leftarrow cEU_q^s.getValue(e)$
10: $\quad u_e \leftarrow minLevenhstein(U(e), e)$
11: $\quad EU_q^s \leftarrow EU_q^s \cup \{(e, u_e)\}$
12: $EU_q^c \leftarrow \emptyset, \quad E_q^c \leftarrow EU_q^s.keys() \cap EU_q^d.keys()$
13: **for all** $e \in E_q^c$ **do**
14: $\quad u_s \leftarrow EU_q^s.getValue(e), \quad u_d \leftarrow EU_q^d.getValue(e)$
15: $\quad u_{sel} \leftarrow maxJaccard(suffix(u_s), suffix(u_d), qa\_obj.q)$
16: $\quad EU_q^c \leftarrow EU_q^c \cup (e, u_{sel})$
17: $E_q^{cr} \leftarrow EU_q^c.keys() \setminus E_q^c$
18: **for all** $re \in E_q^{cr}$ **do**
19: $\quad EU_q^c \leftarrow EU_q^c \cup (re, EU_q^c.getValue(re))$
20: $E_q^{sr} \leftarrow EU_q^s.keys() \setminus E_q^c$
21: **for all** $re \in E_q^{sr}$ **do**
22: $\quad EU_q^c \leftarrow EU_q^c \cup (re, EU_q^s.getValue(re))$
23: **return** EU_q^c

Then we use the fct service for retrieving candidate triples. In our example, we have a single entity, therefore, the fct function will take the following parameters: fct(dbp:Kyoto,population). Another example is the query "Is Nintendo located in Kyoto?", which has 2 entities, i.e., $Nintendo$ and $Kyoto$, thereby, it can use either (i) fct(dbp:Kyoto, {located, dbp:Nintendo}) or (ii)fct(dbp:Nintendo, {located, dbp:Kyoto}). Since LODsyndesis stores such a triple in the entry of both entities, it can return the correct answer regardless of using scenario (i) or (ii). The fct service focuses only to the triples of the selected entity, i.e., $triples(u_{sel}) = \{(s,p,o) \in T | s = u_{sel} \ or \ o = u_{sel}\}$, where $triples(u_{sel}) \subset T$. From those triples, we focus on finding the triples that contain all, or some of the W_q^o words with respect to a given threshold tr. Let $substr(w)$ be all the words that are substrings of a word w, and let c be a single character of a word w, i.e., $substr(w) = \{w' \in W \mid w = c_1 \dots c_n \ and \ w' = c_{1+i} \dots c_{m+i}, \ where \ i \geq 0 \ and \ m+i \leq n\}$. We define as $occur(W_q^o, t, u_{sel})$, the words that can be found in the substring of at least one of the three parts of a triple $t \in triples(u_{sel})$, i.e., a subject, a predicate, or an object: $occur(W_q^o, t, u_{sel})) = \{w \in W_q^o \mid w \in substr(s) \ \ or \ w \in substr(p) \ or \ w \in substr(o)\}, t = (s,p,o) \in triples(u_{sel})\}$. In our running example (see Figures 3.2 and 3.1), the word "population" can be found in two different triples. We define as $score_t$ the ratio of the words W_q^o which are contained in triple t, i.e., $score_t(W_q^o, t, u_{sel}) = \frac{|occur(W_q^o, t, u_{sel})|}{|W_q^o|}$. Its range is [0,1], e.g., a value 0 means that we did not find a word $w \in W_q^o$ in such a triple, while a value 1 means that a triple t contains all the "useful" words of the query. Moreover, we define as $cndT(u_{sel}, W_q^o, tr) = \{t \in triples(u_{sel}) | score_t(W_q^o, t, u_{sel}) \geq tr\}$, the triples satisfying a threshold tr. When the fct service fails to find any fact, the system responds with a message "No answer found", e.g., in Fig. 3.2, the query "Which is the host city of Olympic Games 2020?" cannot be answered by the indexed datasets. On the contrary, if it exists at least one candidate triple, we follow for each question type the approach described below (see lines 5-7 of Alg. 3).

Step **AE.3)** ***Final answer extraction***. Here, at first for each matching URI of the entities of q we retrieve the equivalent URIs using the *objectCoreference* service. Thereafter, we keep only the candidate triples, containing the identified entities in their subject or object. For (a) *factoid questions*, we retrieve the top scored triples using the $score_t$ returned by *factChecking* service. Thereafter, we select the final triple which has the maximum provenance datasets. In our example, where $cndT$ = {(Kyoto, population, 1,474,570,{D_1,D_2}), (Kyoto, population, 1,500,000,{D_3})}. Both triples contain all the question words $W_q^o = \{population\}$ i.e. $score_t = 1$. However, we return as an answer the first one, since it is included in two sources.
For (b) *confirmation questions*, for each matching URI of the entities of q, we retrieve the equivalent URIs using the *objectCoreference* service. Thereafter, for each candidate triple, we try to match all identified entities by checking their subject and object. If a triple exists with all the entities matched, then the system

answer is Yes, otherwise, the answer is No.
Finally for (c) *definition questions*, for extracting the definition of an entity, we check in a predefined set of relations i.e. $definition_{rel}$ = {comment, description, abstract} to extract the desired information. We check each relation in the presented order until we find an answer, i.e., see lines 8-11 of Alg. 3.

Procedure 3 The process of the *Answer Extraction* module

Input: A question analysis object qa_obj, a set of pairs (Entity,URIs) EU_q^c
Output: An answer ans

1: $W_q^o \leftarrow qa\_obj.W_q^u \setminus EU_q^c.keys()$
2: $W_q^o \leftarrow W_q^o \cup SCNLP.getLemmas(W_q^u)$
3: $W_q^o \leftarrow W_q^o \cup WordNet.derivedNouns(W_q^o)$
4: $W_q^o \leftarrow W_q^o \cup WordNet.derivedVerbs(W_q^o)$
5: $u_{sel} \leftarrow EU_q^c.getURIwithMostTriples()$
6: $W_q^o \leftarrow W_q^o \cup (EU_q^c.values() \setminus u_{sel})$
7: $cndT(u_{sel}, W_q^o) \leftarrow fct(u_{sel}, W_q^o)$
8: **if** $cndT(u_{sel}, W_q^o) \neq \emptyset$ **then**
9: $\quad ans \leftarrow cndT(u_{sel}, W_q^o)_{best}$
10: **else**
11: $\quad ans \leftarrow$ "No answer Found"
12: **return** ans

3.1.3 Elaborating on Problematic Process Tasks

We conducted a first evaluation over the SimpleQuestions(v2) dataset [7], in order to get insights about the effectiveness of our approach. We analyzed 400 questions, where 300 of them could not be answered. We identified various problematic cases leading to inability of the system to retrieve any candidate answers. The weaknesses concern two phases of the process:

Named Entity Identification. The problems that arise here are mainly three: 1) the SCNPL fails to recognize the correct entities, 2) the SCNPL fails to recognize any entity, 3) the system fails to retrieve the corresponding URIs of the recognized entities using the *keyword-to-URI* service.

Candidate Triples Retrieval. The problems that arise here, leading to an empty set of candidate triples, are mainly two: 1) there are no available useful words, 2) the related information in the KB is in different phrasing i.e., lexical gap.

Therefore, we decided to elaborate on these two tasks for improving their effectiveness.

3.1.3.1 Improving Entity Identification

As regards *incorrect NER*, we noticed that SCNLP usually fails to recognize correctly multi-word entities, e.g., in the question *what city is vancouver millionaires*

from?, it identifies the entity *Vancouver* instead of *Vancouver Millionaires*, leading to a partial recognition of the entity names. In order to improve the recognition of such entities, we exploit the *Basic dependencies parser* provided by SCNLP for extracting the semantic dependencies in the input question. More specifically, we exploit the *compound* type of relation, between the question words, for being able to capture multi-word entities. In our example, the SCNLP extracts the compound relation *compound(Vancouver, Millionaires)* and therefore we replace the recognized entity name *Vancouver* with its compound version *Vancouver Millionaires*.

As regards all the problematic cases, for improving the recall and the accuracy of the NE Recognition and Linking step, we decided to exploit also the Named Entity Recognition, Disambiguation and Linking capabilities offered by DBpedia Spotlight [31].

Therefore, for finding the "optimal" way to combine the SCNLP and DBpedia Spotlight tools, we conducted a comparative evaluation of three variations using the two tools, presented and discussed in §4.3.1.

3.1.3.2 Improving Triples Retrieval

As regards *no available useful words*, we could exploit external tools such as paraphrasing tools for retrieving a paraphrased version of the input question which contains useful/meaningful words. However, we plan to implement it in future work.

As regards *lexical gap*, for expanding the set of useful words used for the retrieval of candidate triples, we exploit the SCNLP lemmatizer. Specifically, we use the lemmatizer, for extracting a set of the lemmas of the useful words, and then include them in the initial set. Thereafter, for each lemma word, (a) if it is a Verb, we retrieve from the WordNet dictionary (by exploiting the API offered by extJWNL[1]) all the derived nouns, (b) if it is a Noun, we retrieve all the derived verbs and include them in the initial set.

Therefore, for evaluating how each step of the question words expansion, contributes to the task of relevant triples retrieval, and affects the overall results, we conducted a comparative evaluation, presented and discussed in §4.3.1.

[1] `https://github.com/extjwnl/extjwnl`

Chapter 4

Evaluation

This section is organized as follows: §4.1 presents the two evaluation datasets used, namely, SimpleQuestions and QALD-7 Large-Scale dataset. §4.2 presents the evaluation metrics used, which were selected based on the creators of the datasets. Finally, §4.3 presents and discusses the evaluation results.

4.1 Evaluation Datasets used

SimpleQuestions (v2) [7]: It consists of 108,442 simple questions, i.e. questions answerable by a single triple and was built for QA over Freebase KB. It contains $question - triple$ pairs, where *triple* is the corresponding Freebase triple that represents the answer. The evaluation metric used is Accuracy. Note that from the initial set of 108,442 questions, LODsyndesis contains triples for only 3,999 questions. The rest were filtered out. Also note that since LODsyndesis contains information from multiple sources (including DBpedia and Freebase), we may answer a question correctly using DBpedia source but miss the correct matching with the evaluation dataset, due to missing mappings between DBpedia and Freebase. This is a reason that we cannot make a complete and reliable evaluation over this dataset without manually make the aforementioned mappings. Since this task is laborious, time consuming and vulnerable to mistakes, we have randomly extracted 1,000 out of 3,999 questions and manually obtained the correct mappings, to create a final evaluation dataset.

A small set of example questions used, is shown in Table 4.1. While Table 4.2, shows statistics regarding the questions.

QALD-7 [51]: It contains 4 tasks: (a) Multilingual question answering over DBpedia, (b) Hybrid question answering, (c) Large-Scale Question answering over RDF, (d) Question answering over Wikidata. The evaluation metrics used are: mirco and macro Precision, Recall and F1-measure as detailed in [51]. Note that from those 4 tasks, there were participants only in tasks (a) and (d). However, these tasks focus on complicated questions that either require aggregation functions

Question
Which books does the character jessica wakefield appear in?
Who was a victim of the Columbine High School massacre?
What television genre is the program The Night Strangler?
Where is Adler School of professional psychology located?
What is the name of a game that atari published?
What is an event that took place at county louth?
What city is Vancouver Millionaires from?
What films are directed by Brian Blessed?
Who was an advisor for Irving Langmuir?
Who was walter mischel influenced by?
What is the religion of ayesha takia?
What are heath high school (ohio)'s colors?
What is an example of victorian architecture?
What is Sigmund Groven's gender?
Which language is procesado 1040 filmed in?

Table 4.1: Sample questions from SimpleQuestions(v2) dataset.

Variation	Avg Words	Min Words	Max Words
With Stopwords	7.2	3	15
Without Stopwords	4	1	9

Table 4.2: Statistics regarding the 1000 questions from SimpleQuestions(v2).

like greater than (>) and *SUM* (e.g. *"Which countries have **more than** two official languages?"*) or entity-class relations (e.g. *"In which **films** directed by Garry Marshall was Julia Roberts starring?"*) to be answered and are not applicable to a large scale index like the one of LODsyndesisṪo this end we have randomly extracted 2500 simple questions from task (c) (e.g. *"Who was the successor of Amelia Gordon?"*) and compared the answers retrieved from the SPARQL query that retrieves the correct answer wrt the aforementioned benchmark on DBpedia live, with our systems answers. The reason of choosing DBpedia live for comparison is that both LODsyndesis and DBpedia live are publicly web accessible large-scale RDF datasets. Note that due to miss-matches between DBpedia-based answers and other data sources contained in LODsyndesis we had to manually judge the correctness of LODQA answers in case their provenance i.e. RDF dataset, was different from DBpedia. For example for the question *"Who was the successor of Amos Hutchinson?"* our system responses with the literal *"thomas tangretti"* while DBpedia live with the resource *"dbr:Thomas_Tangretti"* which is the same answer so we marked it as true positive. Cases where our system answered with a resource that was not contained in DBpedia answer was marked as false positive, since it is not feasible to manually search whether the answer was correct but not contained in DBpedia live or our system failed to answer the respective question.

A small set of example questions used, from the task (c) of QALD 7 (Large-Scale Question answering over RDF) is shown in Table 4.3. While Table 4.4, shows statistics regarding the questions.

Question
What is the area code of AB postcode area?
Who were the parents of Alexander Helios?
Who was the successor of Yaacob Ibrahim?
In which country is the Three Bears Lake?
Who is the editor of Narrative Magazine?
When did Zalman Shimon Dworkin die?
Who is the founder of Dell Publishing?
When was the Union House built?
What is the capital of Greece?
Who developed Zotero?

Table 4.3: Sample questions from Large-Scale Question answering over RDF QALD 7 dataset.

Variation	Avg Words	Min Words	Max Words
With Stopwords	6.6	3	12
Without Stopwords	4	1	10

Table 4.4: Statistics regarding the 2500 questions from QALD 7 dataset.

4.2 Evaluation Metrics used

The evaluation metrics used for the SimpleQuestions dataset were selected based on the proposed metrics in [7]. Below are namely the metrics along with a description:

(1) Accuracy: Measures the fraction of the correct classifications based on a case

Specifically, regarding the *Entities Detection step*, the accuracy is calculated as the percentage of the questions where the Named Entity Recognition and Linking was performed correctly. Regarding the *Answer Extraction step*, the accuracy is calculated as the percentage of the questions correctly answered by LODQA.

The evaluation metrics used for the QALD dataset were selected based on the proposed metrics of QALD-7. Below are namely the metrics along with a description based on [14, 51]:

Micro: Micro-average aggregates the contributions of the answers of all questions to compute the average metric.

Macro: Macro-average computes the metric independently for each question and then average them, hence it treats all questions equally.

(1) Micro/Macro Precision: Measures for a single question, the number of correct system answers, divided by the number of system answers.

(2) Micro/Macro Recall: Measures for a single question, the number of correct system answers, divided by the number of the gold answers.

(3) Micro/Macro F-measure: Measures for a single question, the weighted harmonic mean of Precision and Recall

4.3 Evaluation Results

This section presents and discusses the evaluation results over the two datasets used. Specifically, §4.3.1 presents the results over the SimpleQuestions dataset, aiming to optimize the problematic tasks of the QA process (a) Entities Detection and (b) Answer Extraction. While §4.3.2 presents the results over the QALD-7 Large-Scale dataset, aiming to get insights about the effectiveness of LODQA as well as for reflecting how each step of the process affects the overall results.

Method	Accuracy
SCNLP-Spotlight	0.626
Spotlight-SCNLP	0.653
Combined	**0.737**

Table 4.5: Accuracy of each Named Entity Recognition approach over 1000 questions on SimpleQuestions(v2).

4.3.1 Evaluation results over SimpleQuestions(v2)

For being able to optimize the QA process as a whole and specifically the two problematic process tasks namely: *Entities Detection* and *Answer Extraction*, discussed and detailed in §3.1.3, we used a subset of 1000 questions from the SimpleQuestion (v2) dataset.

Regarding the task of *Entities Detection*, the different models used in the evaluation, for finding the "optimal" way to combine the SCNLP and DBpedia Spotlight tools, are the following three:

1. *SCNLP-Spotlight:* Perform Named Entity recognition (NER) using the SCNLP and Named Entity linking (NEL) using the *keyword-to-URI* service. In case any of these fails, then perform NER and NEL using the DBpedia Spotlight in the input question.

2. *Spotlight-SCNLP:* Perform Named Entity recognition and Named Entity Linking using the DBpedia Spotlight in the input question. In case of failure, use the SCNLP for the NE recognition and *keyword-to-URI* service for the NE linking.

3. *Combined:* Perform Named Entity recognition and Named Entity Linking using both SCNLP and DBpedia Spotlight tools. Then, based on simple heuristics, select the best entities.

We measured the accuracy of these three approaches using the 1000 simple questions and the results are shown in Table 4.5.

We observe that the combined method achieves the highest accuracy (0.73). By inspecting the individual results, we noticed that, (1) DBpedia Spotlight performs better in cases where (a) NE disambiguation is required, since SCNLP does not include a NE disambiguation step, and (b) the corresponding question entity in the underlying KB has different label, e.g. in the question *what is the film tempo di uccidere about*, spotlight identifies {tempo di uccidere = http://dbpedia.org/resource/Time_to_Kill_(1989_film)}. (2) SCNLP performs better in cases where (a) recognition of multi-word entities is required, and (b) recognition of common type of entities i.e., person, location, organization is required.

Method	Accuracy	
	Total	Perfect ED
LODQA	**0.487**	**0.642**
LODQA w/o L	0.411	0.556
LODQA w/o N	0.414	0.558
LODQA w/o V	0.429	0.581
LODQA w/o L,N,V	0.407	0.547

Table 4.6: Accuracy of each Triples Retrieval approach over (a) 1000 questions on SimpleQuestions(v2) and (b) a subset of these questions where the *Entities Detection* step was successful.

Regarding the task of *Answer Extraction*, the different variations used in the evaluation, for understanding how each step of the question words expansion, contributes to the task of relevant triples retrieval, as well as, how it affects the final results, are the following:

1. *LODQA:* Our approach by using all the expansion steps (i.e., lemmas, nouns, verbs) presented and detailed in §3.1.2.
2. *LODQA w/o L:* A variation of our approach which does not use lemmas
3. *LODQA w/o N:* A variation of our approach which does not perform expansion based on nouns
4. *LODQA w/o V:* A variation of our approach which does not perform expansion based on verbs
5. *LODQA w/o L,N,V:* A variation of our approach which does not perform any expansion over the available question words

We measured the precision as defined in [8] over our systems answers on the final evaluation dataset described in 4.1. Table 4.6 shows the results.

The keynote here, is that LODQA achieves the highest accuracy (0.49), which reflects the importance of tackling the lexical gap between the input question and the underlying sources. Moreover, it seems that each part-of-speech plays its own important role in the Triples Retrieval process. For this dataset we report that (i) verbs are more important that nouns and lemmas and (ii) when all part-of-speech are involved in the answer extraction process we succeed gain of (0.08) in terms of accuracy. In addition, since *Entities Detection* affects *Answer Extraction* it is reasonable to evaluate the latter module independently of the first. To this end we evaluated our approaches only over the portion of questions that *Entities Detection* was successful. That results to a dataset consisting of 737 questions in total. We then measured the Accuracy over those questions. The results are shown in the last column of table 4.6. The observations in this scenario totally agree with the report over the total 1,000 questions.

4.3.2 Evaluation results over QALD Large-Scale dataset

For getting insights about the effectiveness of our approach as a whole as well as for reflecting how each step in the proposed pipeline affects the overall results, we extracted and used 2500 questions from the Large-Scale Question answering over RDF (QALD-7) dataset.

To the best of our knowledge there is no other work evaluated on the same collection and the same subset, for this reason we comparatively evaluate variations of our proposed approach for evaluating how each step affects the final result i.e. the final answer and discuss the results.

The different models (i.e. variations of our approach) used in the evaluation, were extracted by creating variations of two modules: *Entities Detection* and *Answer Extraction*. Concerning the *Entities Detection* step, the models used in the evaluation are listed below and they have the following characteristics. 1) LODQA+ Stanford Corenlp 2) LODQA+ DBpedia Spotlight 3) LODQA (i.e., it uses both DBpedia Spotlight and Stanford Corenlp). The first model uses only the SCNLP for this task, the second model uses only DBPedia Spotlight, while the third one exploits both tools in a combined method.

As regards the *Answer Extraction*, we evaluate the effectiveness of our approach by using all the expansion steps (i.e., lemmas, nouns, verbs) presented in §3.1.2. Moreover, we compare that approach with variations that do not perform expansion based on: a) lemmas (LODQA w/o L) i.e. we do not include the lemmas of the question words, b) nouns (LODQA w/o N) i.e. we do not extract derived verbs based on nouns, c) verbs (LODQA w/o V) i.e. we do not extract derived nouns based on verbs and d) all the aforementioned ones (LODQA w/o L,N,V), i.e., a variation of our approach, where the expansion of the available words with their lemmas and derived nouns and verbs is not applied (it does not consider the lexical gap between the question and the underlying sources).

The evaluation results are shown in Table 4.7, and in Figures 4.1, 4.2, 4.3.

Metric/Model	Mi-Precision	Mi-Recall	Mi-F1	Ma-Precision	Ma-Recall	Ma-F1
LODQA + SCNLP	0.456	0.279	0.346	0.462	0.231	0.308
LODQA + Spotlight	0.559	0.328	0.413	0.575	0.283	0.379
LODQA	0.616	**0.351**	**0.447**	**0.637**	**0.314**	**0.421**
LODQA w/o L	0.610	0.347	0.442	0.629	0.310	0.415
LODQA w/o N	**0.617**	0.350	**0.447**	**0.637**	**0.314**	**0.421**
LODQA w/o V	0.587	0.337	0.428	0.604	0.298	0.399
LODQA w/o L,N,V	0.534	0.333	0.410	0.596	0.294	0.394

Table 4.7: Evaluation results over XX question of QALD-7 dataset

As we can see from the results in Table 4.7, the full version (i.e. LODQA) and the one that does not consider any expansion based on nouns (i.e. LODQA w/o N) of our proposed approach outperform all the other models in all metrics achieving $Macro - Recall = 0.314$, $Macro - Precision = 0.637$, $Micro - F1 = 0.447$ and

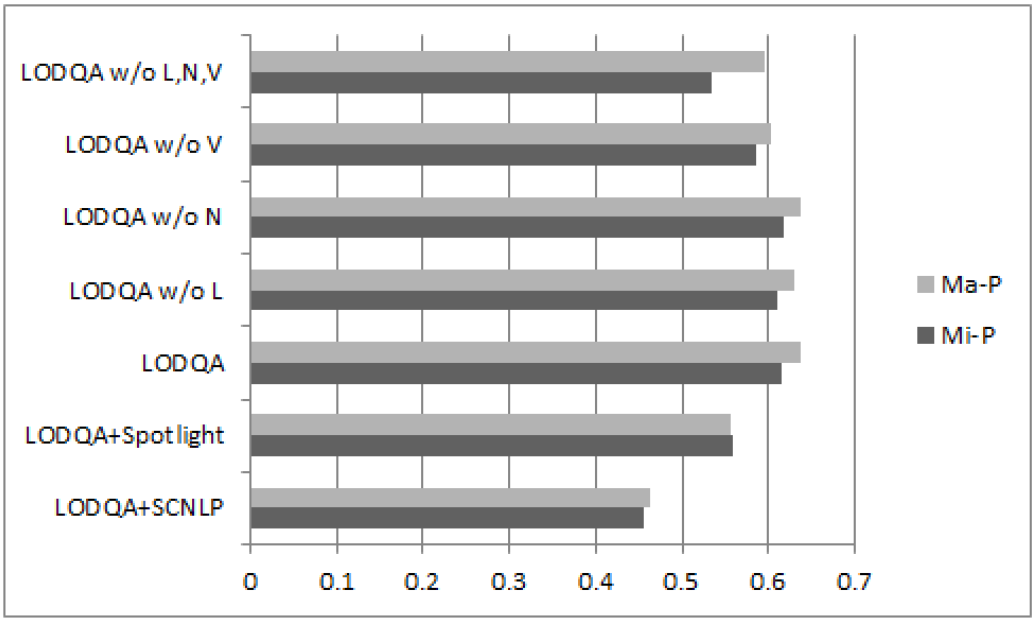

Figure 4.1: Micro and Macro Precision of the models over QALD-7 dataset.

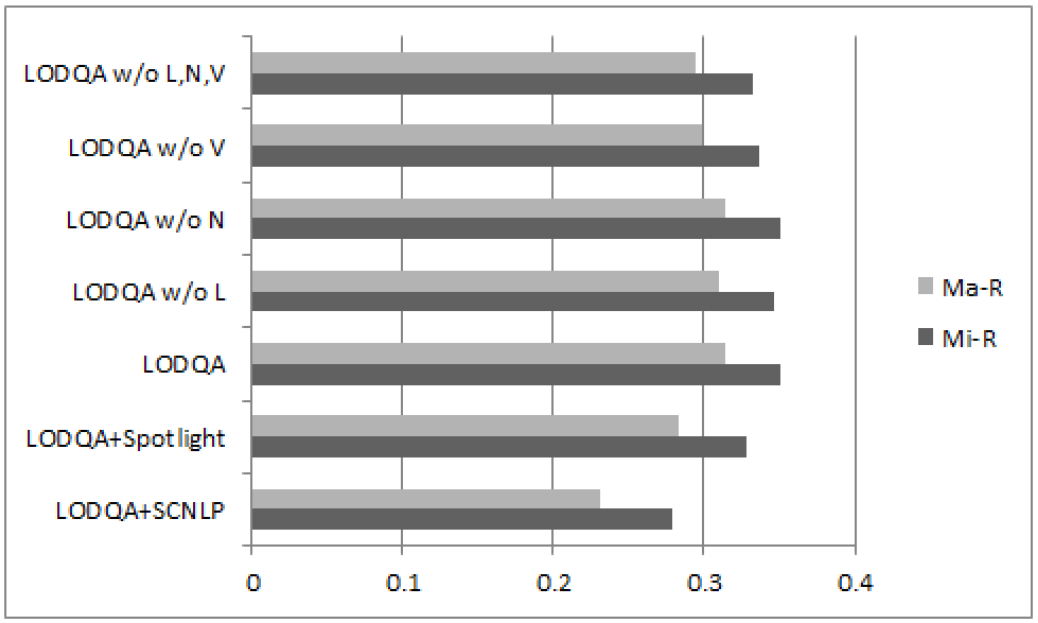

Figure 4.2: Micro and Macro Recall of the models over QALD-7 dataset.

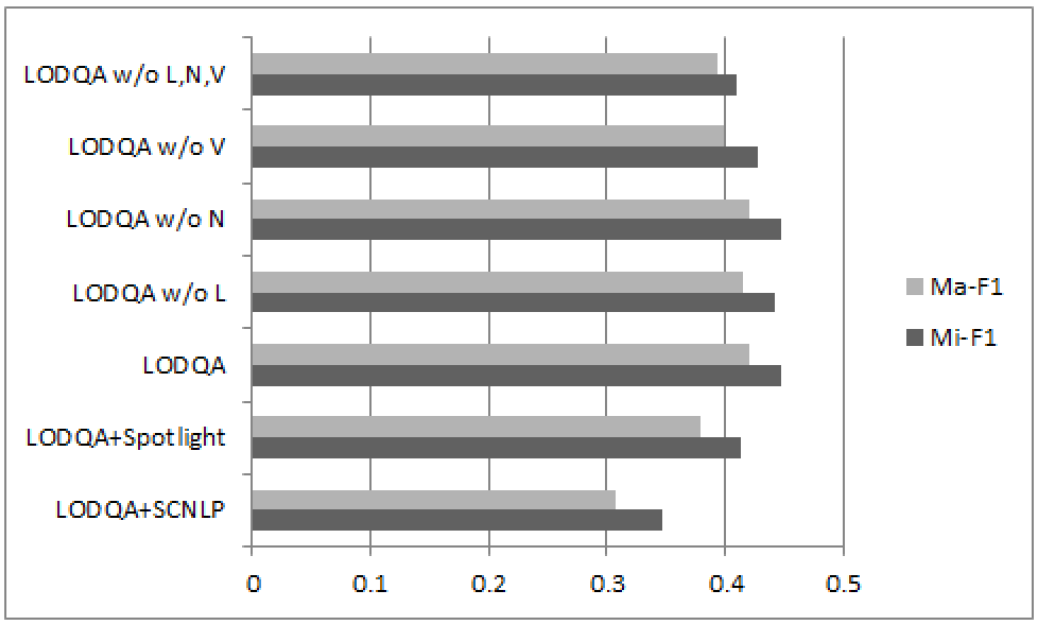

Figure 4.3: Micro and Macro F1 of the models over QALD-7 dataset.

$Macro - F1 = 0.421$. The difference between those two models is that LODQA slightly outperform LODQA w/o N at $Micro - Recall$ (0.351 over 0.350), while LODQA w/o N slightly outperform LODQA at $Micro - Precision$ (0.617 over 0.616).

Comparison of Approaches for Entities Detection step. We can observe that our combined method using both Stanford CoreNLP and DBPedia Spotlight for the recognition and linking of the Named Entities, outperforms all the other approaches, achieving more accurate results. More specifically, it gains an improvement of 0.034 in terms of $Micro - F1$ and 0.031 in terms of $Macro - F1$ over the next best performing model (LODQA + Spotlight) in the overall QA process (see Table 4.7) and an improvement of 0.084 in terms of accuracy over the next best performing model LODQA + Spotlight in the Named Entity linking step.

Comparison of Approaches for Answer Extraction step. Another keynote here, is that without the expansion of the available words with their lemmas and derived nouns and verbs (i.e., for tackling the lexical gap) we achieve lower results which indicates the importance of tackling the lexical gap between the input question and the underlying sources for retrieving the correctly relevant information (see method LODQA in table 4.7). More specifically, it seems that nouns is the most important part of speech for our QA task, since method LODQA w/o N, i.e. the method that retrieves the derived nouns from the verb question terms, achieves results similar to LODQA (which applies the same expansion and vice versa) and outperforms all the other expansion variations. That means that

method LODQA w/o N tends to result to a list of useful words with more nouns than verbs and that seems to have a positive impact on the final Answer Extraction. Moreover, we report that lemmas (which is not considered a part of speech) are quite important too. This is observable from LODQA w/o L, which also outperforms LODQA w/o V. This is quite reasonable, since lemmas are usually the smallest set of characters (not always though e.g. *best* has as lemma *good* which has the same number of characters) that keep the meaning of a word and thus matches more often with parts of the candidate relations.

Chapter 5

Implementation and Applications

5.1 Implementation

This thesis, has been implemented using the Java programming language and version 8 as well the editor NetBeans IDE version 8.2. The code is web accesible from the following link[1].

Below there is a list of tools used in the proposed QA process:

1. Stanford CoreNLP [2] version 3.8.0.

2. WordNet Lexicon [3] version 3.1.0.

3. Extended Java WordNet Library (extJWNL) [4] version 1.9.4.

The SCNLP tool was exploiting for the Named Entity Recognition capabilities as well as the offered lemmatizer and semantic/syntactic analysis. The WordNet dictionary was exploited for the expansion of the question words and the Extended Java WordNet Library was exploited due to the offered functionality to extract derivations of words based on their POS tags.

Below there is a list of tools used for tasks related to the QA process:

1. GSON Library [5] version 2.8.1.

2. Apache Jena [6] version 3.9.0.

[1] https://github.com/SemanticAccessAndRetrieval/QuestionAnswering
[2] https://stanfordnlp.github.io/CoreNLP/
[3] https://wordnet.princeton.edu/
[4] https://github.com/extjwnl/extjwnl
[5] https://github.com/google/gson
[6] https://jena.apache.org/

The GSON Library was exploited to easily read and extract content from json objects, as well as for communicating with the LODsyndesis services. While, Apache Jena was used for constructing the final evaluation collection over the QALD-7 Large-Scale dataset.

5.2 Applications and Applicability

The approach described in this thesis has been implemented in the context of a research prototype LD-SDS system [39] that is developed by TOSHIBA Research Europe and FORTH. Which is a system that combines spoken dialogue and faceted search and is limited to spoken dialogues over structured datasets. In Figure 5.1 we can see the general context.

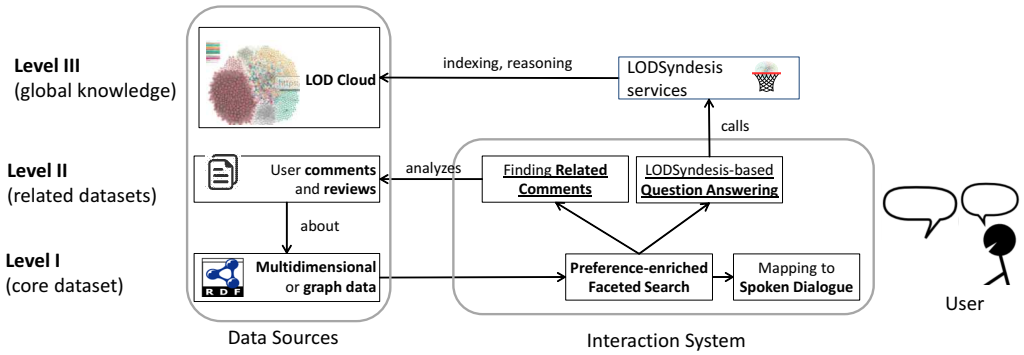

Figure 5.1: Exploiting External Source in Spoken Dialogue Faceted Search

While Figure 5.2 shows the main technical components of the LD-SDS architecture.

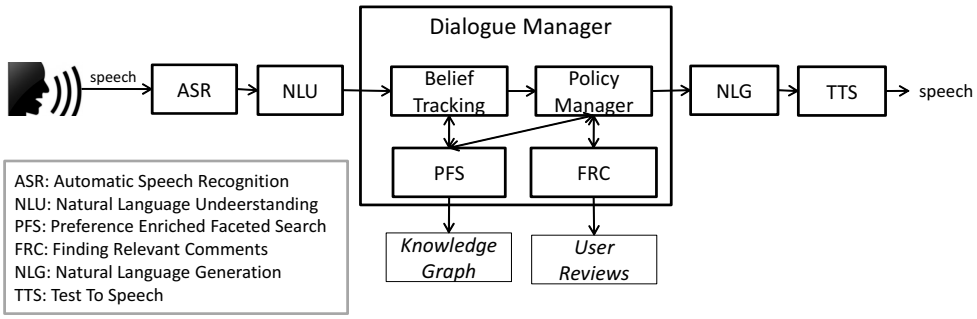

Figure 5.2: Architecture of the LD-SDS System

This thesis has been implemented as a standalone software component that can be exploited from other applications and services, as well as directly from users. As regards the former, we have integrated it with a question answering software component, we have developed that is capable of answering questions from a corpus of many short excerpts (e.g. user comments or reviews). The

rationale for integrating these components is that it will allow users searching over the corpus of the particular context, and if there are no relevant information then the open domain QA system will be used for answering the question. As regards the latter, it has implemented as a RESTful service that accept user questions.

In order to enhance the user experience, we are currently enhancing it with modules that will enable speech interaction. More specifically we aim to enhance it with speech-2-text and text-2-speech facilities so that users will be able to use it in an intuitive manner.

A demo application of this thesis can be found in the following link[7]. Figure 5.3, shows the welcome page of the demo, where the user has two options: (i) to try the service, and submit a question he/she wishes for discovering the desired answer, or (ii) to check a set of demo queries, for the three question types.

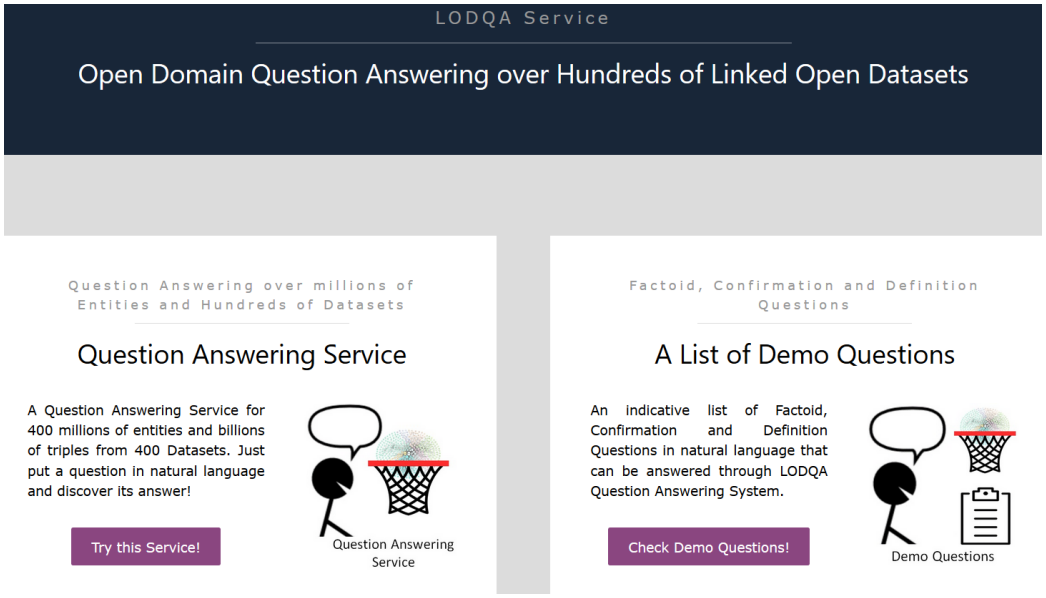

Figure 5.3: Demo Application: Welcome page.

Figure 5.4, shows the page where the user is able to submit his/her question, and select the format of the answer (i.e. (i) plain text, or (ii) triple format). While Figure 5.5 shows the answer of the question *Which was the birth place of Socrates?*, as well as the complete question analysis, where the user has also the option to further explore via the LODsyndesis services, the detected entities.

Finally, Figure 5.6, shows the page containing a set of demo queries for each question type.

[7] `http://83.212.101.193:8080/LODQA/`

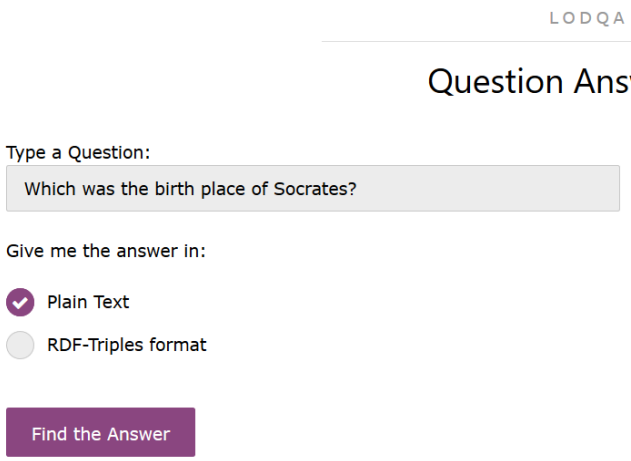

Figure 5.4: Demo Application: Submit your question.

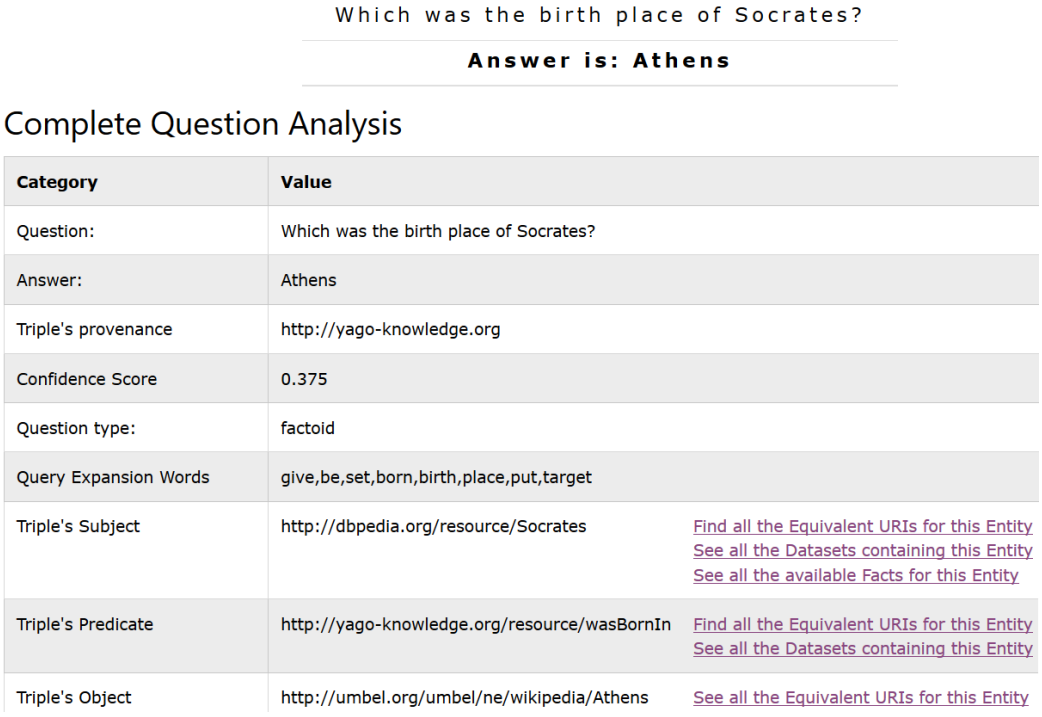

Figure 5.5: Demo Application: Answer to an example query.

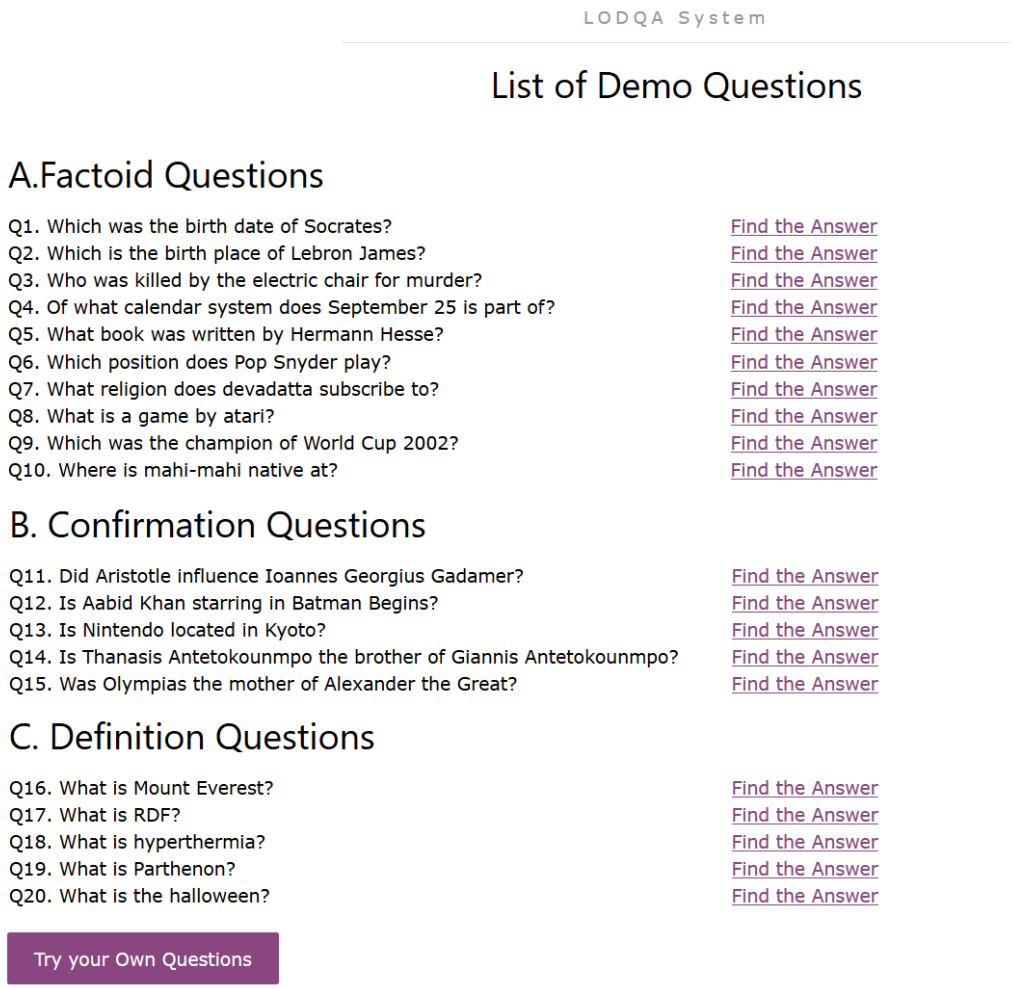

Figure 5.6: Demo Application: Demo queries page.

Chapter 6

Conclusion

6.1 Concluding Remarks

We have described an information extraction-based approach for open domain QA that exploits the wealth of data coming from hundreds of datasets and does not depend on the availability of training data. The multiplicity of datasets allows verifying an answer to a given question from multiple sources, and it increases the number of answerable questions. In comparison to related systems like WDAqua [15], AMAL [41], Aqqu [3], [61], and SINA [45], where a single or few KBs are supported, our approach exploits the contents of 400 datasets enriched with all inferred equivalence relationships.

We demonstrated the benefits of this approach in terms of answerable questions and answer verification. We have investigated how the steps of the QA process affect the effectiveness of QA, specifically as regards the entity linking process, the combination of SCNLP and DBpedia Spotlight increases the performance in entity linking achieving an accuracy of 0.737 with a gain of 0.084 over Spotlight in the SimpleQuestions (v2) dataset and in the overall QA process achieving a $Micro-F1$ of 0.447 and $Macro-F1$ of 0.421 over with a gain of 0.005 and 0.004 over LODQA w/o L in the QALD-7 dataset. As concerns the answer extraction, we report that nouns have the greatest impact when expanding the useful words since LODQA and LODQA w/o N performs similarly. We have reported experimental results over 1000 question-answer pairs from SimpleQuestions (v2) for the entity linking step and over 2500 question from the QALD-7 dataset for the over QA process. The results show that without expanding the useful words the retrieval of the triple answer performs poorly and that without using both tools that are specialized for the large datasets of LODsyndesis (i.e. Spotlight - DBpedia) and more generic tools that are KB agnostic (i.e. SCNLP) the performance of entity linking and as such of QA does not perform adequately.

Issues that are worth further research include the extension of LODQA for supporting more question types e.g. list questions, which is appropriate for our case since we exploit complementary information from multiple sources. Investigate

more efficient ways to tackle the *lexical gap* issue for the tasks of relevant triples retrieval and relation matching.

Bibliography

[1] Abdalghani Abujabal, Mohamed Yahya, Mirek Riedewald, and Gerhard Weikum. Automated template generation for question answering over knowledge graphs. In *Proceedings of the 26th international conference on world wide web*, pages 1191–1200. International World Wide Web Conferences Steering Committee, 2017.

[2] Ali Mohamed Nabil Allam and Mohamed Hassan Haggag. The question answering systems: A survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS)*, 2(3), 2012.

[3] Hannah Bast and Elmar Haussmann. More accurate question answering on freebase. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1431–1440. ACM, 2015.

[4] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, volume 2, page 6, 2013.

[5] Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. In *ACL (1)*, pages 1415–1425, 2014.

[6] Jonathan Berant and Percy Liang. Imitation learning of agenda-based semantic parsers. *Transactions of the Association for Computational Linguistics*, 3:545–558, 2015.

[7] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075, 2015.

[8] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*, 2015.

[9] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051, 2017.

[10] Zihang Dai, Lei Li, and Wei Xu. Cfo: Conditional focused neural question answering with large-scale knowledge bases. *arXiv preprint arXiv:1606.01994*, 2016.

[11] Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. Question answering on knowledge bases and text using universal schema and memory networks. *arXiv preprint arXiv:1704.08384*, 2017.

[12] Manmita Devi and Mohit Dua. Adans: An agriculture domain question answering system using ontologies. In *Computing, Communication and Automation (ICCCA), 2017 International Conference on*, pages 122–127. IEEE, 2017.

[13] Mamadou Diao, Sougata Mukherjea, Nitendra Rajput, and Kundan Srivastava. Faceted search and browsing of audio content on spoken web. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1029–1038. ACM, 2010.

[14] Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information systems*, 55(3):529–569, 2018.

[15] Dennis Diefenbach, Kamal Singh, and Pierre Maret. Wdaqua-core1: A question answering service for rdf knowledge bases. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pages 1087–1091. International World Wide Web Conferences Steering Committee, 2018.

[16] Eleftherios Dimitrakis, Konstantinos Sgontzos, Panagiotis Papadakos, Yannis Marketakis, Alexandros Papangelis, Yannis Stylianou, and Yannis Tzitzikas. On finding the relevant user reviews for advancing conversational faceted search. In *Proceedings of 4th Workshop on Sentic Computing, Sentiment Analysis, Opinion Mining, and Emotion Detection (EMSASW 2018) Co-located with the 15th Extended Semantic Web Conference 2018 (ESWC 2018), Heraklion, Greece, June 4, 2018.*, pages 22–31, 2018.

[17] P. Fafalios and Y. Tzitzikas. X-ENS: Semantic Enrichment of Web Search Results at Real-Time. In *SIGIR'13*, pages 1089–1090, Dublin, Ireland, 2013.

[18] Pavlos Fafalios, Michail Salampasis, and Yannis Tzitzikas. Exploratory patent search with faceted search and configurable entity mining. In *1st International Workshop on Integrating IR technologies for Professional Search (ECIR'13 Workshop)*, 2013.

[19] Yansong Feng, Songfang Huang, Dongyan Zhao, et al. Hybrid question answering over knowledge base and free text. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2397–2407, 2016.

[20] Sébastien Ferré. Sparklis: an expressive query builder for sparql endpoints with guidance in natural language. *Semantic Web*, 8(3):405–418, 2017.

[21] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.

[22] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[23] Sherzod Hakimov, Soufian Jebbara, and Philipp Cimiano. Amuse: multilingual semantic parsing for question answering over linked data. In *International Semantic Web Conference*, pages 329–346. Springer, 2017.

[24] Shizhu He, Shulin Liu, Yubo Chen, Guangyou Zhou, Kang Liu, and Jun Zhao. Casia@ qald-3: A question answering system over linked data. In *CLEF (Working Notes)*, 2013.

[25] Konrad Höffner, Sebastian Walter, Edgard Marx, Ricardo Usbeck, Jens Lehmann, and Axel-Cyrille Ngonga Ngomo. Survey on challenges of question answering in the semantic web. *Semantic Web*, 8(6):895–920, 2017.

[26] Panagiotis Lionakis and Yannis Tzitzikas. Pfsgeo: Preference-enriched faceted search for geographical data. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 125–143. Springer, 2017.

[27] Hongxia Liu, Qingcheng Hu, Yong Zhang, Chunxiao Xing, and Ming Sheng. A knowledge-based health question answering system. In Hsinchun Chen, Daniel Dajun Zeng, Elena Karahanna, and Indranil Bardhan, editors, *Smart Health*, pages 286–291, Cham, 2017. Springer International Publishing.

[28] Betia Lizbeth López-Ochoa, José Luis Sánchez-Cervantes, Giner Alor-Hernández, Ma Antonieta Abud-Figueroa, Beatriz A Olivares-Zepahua, and Lisbeth Rodríguez-Mazahua. An architecture based in voice command recognition for faceted search in linked open datasets. In *International Conference on Software Process Improvement*, pages 174–185. Springer, 2017.

[29] Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. Neural network-based question answering over knowledge graphs on word and character level. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1211–1220. International World Wide Web Conferences Steering Committee, 2017.

[30] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.

[31] Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems*, pages 1–8. ACM, 2011.

[32] Florian Metze, Xavier Anguera, Etienne Barnard, Marelie Davel, and Guillaume Gravier. The spoken web search task at mediaeval 2012. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8121–8125. IEEE, 2013.

[33] Amit Mishra and Sanjay Kumar Jain. A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, 28(3):345–361, 2016.

[34] Diego Mollá and José Luis Vicedo. Question answering in restricted domains: An overview. *Comput. Linguist.*, 33(1):41–61, March 2007.

[35] Michalis Mountantonakis and Yannis Tzitzikas. On measuring the lattice of commonalities among several linked datasets. *Proceedings of the VLDB Endowment*, 9(12), 2016.

[36] Michalis Mountantonakis and Yannis Tzitzikas. High performance methods for linked open data connectivity analytics. *Information*, 9(6):134, 2018.

[37] Panagiotis Papadakos, Nikos Armenatzoglou, Stella Kopidaki, and Yannis Tzitzikas. On exploiting static and dynamically mined metadata for exploratory web searching. *Knowledge and information systems*, 30(3):493–525, 2012.

[38] Panagiotis Papadakos and Yannis Tzitzikas. Comparing the effectiveness of intentional preferences versus preferences over specific choices: a user study. *International Journal of Information and Decision Sciences*, 8(4):378–403, 2016.

[39] Alexandros Papangelis, Panagiotis Papadakos, Margarita Kotti, Yannis Stylianou, Yannis Tzitzikas, and Dimitris Plexousakis. Ld-sds: Towards an expressive spoken dialogue system based on linked-data. In *Search Oriented Conversational AI, SCAI 17 Workshop (co-located with ICTIR 17)*, 2017.

[40] Seonyeong Park, Soonchoul Kwon, Byungsoo Kim, and Gary Geunbae Lee. Isoft at qald-5: Hybrid question answering system over linked data and text data. In *CLEF (Working Notes)*, 2015.

[41] Nikolay Radoev, Mathieu Tremblay, Michel Gagnon, and Amal Zouaq. Answering natural language questions on rdf knowledge base in french. *7th open challenge in Question Answering over Linked Data (QALD-7), Portoroz, Slovenia*, 2017.

[42] Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140, 2016.

[43] Giovanni Maria Sacco and Yannis Tzitzikas. *Dynamic taxonomies and faceted search: theory, practice, and experience*, volume 25. Springer Science & Business Media, 2009.

[44] Denis Savenkov and Eugene Agichtein. When a knowledge base is not enough: Question answering over knowledge bases with external text data. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 235–244, New York, NY, USA, 2016. ACM.

[45] Saeedeh Shekarpour, Edgard Marx, Axel-Cyrille Ngonga Ngomo, and Sören Auer. Sina: Semantic interpretation of user queries for question answering on interlinked data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 30:39–51, 2015.

[46] Evgeny Sherkhonov, Bernardo Cuenca Grau, Evgeny Kharlamov, and Egor V Kostylev. Semantic faceted search with aggregation and recursion. In *International Semantic Web Conference*, pages 594–610. Springer, 2017.

[47] Y. Tzitzikas and E. Dimitrakis. Preference-enriched faceted search for voting aid applications. *IEEE Transactions on Emerging Topics in Computing*, PP(99):1–1, 2016.

[48] Yannis Tzitzikas, Nicolas Bailly, Panagiotis Papadakos, Nikos Minadakis, and George Nikitakis. Using preference-enriched faceted search for species identification. *International Journal of Metadata, Semantics and Ontologies*, 11(3):165–179, 2016.

[49] Yannis Tzitzikas, Nikos Manolis, and Panagiotis Papadakos. Faceted exploration of rdf/s datasets: a survey. *Journal of Intelligent Information Systems*, pages 1–36, 2016.

[50] Yannis Tzitzikas and Panagiotis Papadakos. Interactive exploration of multidimensional and hierarchical information spaces with real-time preference elicitation. *Fundamenta Informaticae*, 20:1–42, 2012.

[51] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Bastian Haarmann, Anastasia Krithara, Michael Röder, and Giulio Napolitano. 7th open challenge on question answering over linked data (qald-7). In *Semantic Web Evaluation Challenge*, pages 59–69. Springer, 2017.

[52] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 189–198, 2017.

[53] Yusuke Watanabe, Bhuwan Dhingra, and Ruslan Salakhutdinov. Question answering from unstructured text by retrieval and comprehension. *CoRR*, abs/1703.08885, 2017.

[54] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*, 2016.

[55] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. Question answering on freebase via relation extraction and textual evidence. *arXiv preprint arXiv:1603.00957*, 2016.

[56] Kun Xu, Sheng Zhang, Yansong Feng, and Dongyan Zhao. Answering natural language questions via phrasal semantic parsing. In *Natural Language Processing and Chinese Computing*, pages 333–344. Springer, 2014.

[57] Xuchen Yao, Jonathan Berant, and Benjamin Van Durme. Freebase qa: Information extraction or semantic parsing. In *Proceedings of ACL*, 2014.

[58] Xuchen Yao and Benjamin Van Durme. Information extraction over structured data: Question answering with freebase. In *ACL (1)*, pages 956–966, 2014.

[59] Semih Yavuz, Izzeddin Gur, Yu Su, Mudhakar Srivatsa, and Xifeng Yan. Improving semantic parsing via answer type inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 149–159, 2016.

[60] Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1321–1331, 2015.

[61] Yuanzhe Zhang, Shizhu He, Kang Liu, and Jun Zhao. A joint model for question answering over multiple knowledge bases. In *AAAI*, pages 3094–3100, 2016.

[62] Lei Zou, Ruizhe Huang, Haixun Wang, Jeffrey Xu Yu, Wenqiang He, and Dongyan Zhao. Natural language question answering over rdf: a graph data driven approach. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 313–324. ACM, 2014.