Preprint of the paper:
E. Dimitrakis, K. Sgontzos, M. Mountantonakis, Y. Tzitzikas,
Enabling Efficient Question Answering over Hundreds of Linked Datasets,
Post-proceedings of the 13th International Workshop on
Information Search, Integration, and Personalization (ISIP'2019),
Lecture Notes in Computer Science book series (LNCS, volume 1197), 2020 (to appear)

Enabling Efficient Question Answering over Hundreds of Linked Datasets

Eleftherios Dimitrakis, Konstantinos Sgontzos, Michalis Mountantonakis, and Yannis Tzitzikas

Institute of Computer Science - FORTH-ICS, Greece, and
Computer Science Department - University of Crete, Greece
{dimitrakis, sgontzos, mountant, tzitzik}@ics.forth.gr

Abstract. In this paper we introduce an approach, called LODQA, for open domain Question Answering over Linked Open Data. We confine ourselves to three kinds of questions: factoid, confirmation, and definition questions. By using LODQA it is feasible to answer questions over 400 millions of entities of any domain without using any training data, since we exploit simultaneously 400 Linked datasets. In particular, we exploit the services of LODsyndesis, a suite of services (based on semantics-aware indexes) which supports cross-dataset reasoning over hundreds of Linked datasets and 2 billion triples. The proposed Question Answering process follows an information extraction approach and comprises several steps including question cleaning, heuristic based question type identification, entity recognition, linking and disambiguation using Linked Data-based methods and pure NLP methods (specifically DBpedia Spotlight and Stanford CoreNLP), WordNet-based question expansion for tackling the lexical gap (between the input question and the underlying sources), and triple scoring for producing the final answer. We discuss the benefits of this approach in terms of answerable questions and answer verification, and we investigate, through experimental results, how the aforementioned steps of the process affect the effectiveness and the efficiency of question answering.

Keywords: Questions Answering, Linked Data, Multiple Datasets

1 Introduction

Although the first QA (Question Answering) systems were created decades ago (back in 1960s), the problem is still open since the existing techniques have several limitations (for more see [24]), therefore QA is subject of continuous research. There is a wide range of techniques for QA ranging from simple manually-written regular expression-based methods, to methods relying on deep learning, e.g. see the survey papers [16, 21, 28], and there are several collections for evaluating QA systems (see [8]). Recently we observe a wide adoption of QA-based personal assistants (including Apple's Siri, Google Assistant, Amazon's Alexa) that are capable of answering a wide range of questions, as well as an increasing interest from the database community for natural language interfaces to

databases [2, 26]. Indeed natural language interfaces can complement the existing methods for query formulation by casual users, i.e. faceted search [27], as evidenced by prototypes supporting spoken dialogue interfaces for information navigation [20].

Open domain (as opposed to closed domain) Question Answering (QA) is a challenging task, since it requires to tackle a number of issues: (i) the issue of data distribution, i.e., several datasets, that are usually distributed in different places, should be exploited for being able to support open domain question answering, (ii) the difficulty of word sense disambiguation, because the associated vocabulary is not restricted to a single domain, and (iii) the difficulty (or inability) to apply computationally expensive techniques, such as deep NLP analysis, due to the huge size of the underlying sources. In this paper we focus on Open Domain Question Answering over *Linked Data*. We introduce `LODQA`, a Linked Data-based Question Answering system that exploits `LODsyndesis` [17], a recently launched suite of services over hundreds of LOD Datasets (that contains two billion triples about 400 million entities). We selected to use `LODsyndesis`, because if offers two distinctive features for the QA process, which are not supported by a single source: (a) it is feasible to verify an answer to a given question from several sources, and (b) the number of questions that can be answered is highly increased, because datasets usually contain complementary information for the same topics and entities. Essentially, we try to find the best triple(s) for answering the incoming question; we do not carry out any other information integration techniques (like those surveyed in [19]).

Regarding (a), suppose that the given question is *"What is the population of Heraklion?"*, and the system retrieves two candidate triples, i.e., {(Heraklion, population, 140,730), (Heraklion, population, 135,200)}. The two triples contain a different value for the population of that city, however, suppose that the first triple can be verified from four datasets (say D_1, D_2, D_3, D_4), whereas the second one only from a single dataset (say D_5). In this example, `LODQA` will return as correct answer the first triple, because it can be verified from a larger number of datasets, thereby, we have more evidence about its correctness.

Regarding (b), suppose that `LODQA` receives the two following questions for an other domain (say marine domain), *"Is Yellowfin Tuna a predator of Atlantic pomfret?"* and *"Which is the genus of Yellowfin Tuna?"*. These two questions are addressed to the same entity i.e. *"Yellowfin Tuna"*, however there is not a single dataset where we can find the desired information for answering both questions. Indeed, `LODQA` is able to answer the first question by using data from *Ecoscope* dataset, whereas the second question is answerable by a triple that occur in *DBpedia* knowledge base.

Concerning the Question Answering process followed from `LODQA`, it is an information extraction approach, as opposed to the semantic parsing approach, that consists of multiple steps. In particular, `LODQA` performs question cleaning (e.g., removal of stopwords) and it identifies the question type (e.g., factoid) by exploiting heuristics. Moreover, it recognizes the entities of the question and it performs linking and disambiguation, by using both pure NLP methods and

Linked Data-based methods, specifically Stanford CoreNLP [9,13] and DBpedia Spotlight [14]. Furthermore, it uses *WordNet* [15] for tackling the possible lexical gap between a given question and the answer which can be found in the underlying sources. Finally, it receives the candidate triples from `LODsyndesis`, and it scores each candidate triple for producing the final answer. Concerning evaluation, we discuss the benefits of this approach in terms of answerable questions and answer verification, and we investigate through experimental results, how the aforementioned steps of the process affect the effectiveness and the efficiency of question answering.

The rest of this paper is organized as follows: Section 2 discusses related work, Section 3 introduces the proposed approach, Section 4 reports comparative experimental results, whereas Section 5 describes an application of the proposed approach. Finally, Section 6 concludes the paper and discusses directions for future research and work.

2 Related Work

Knowledge Base Question Answering (KBQA) systems can be divided in two different categories: (a) *Semantic Parsing (SP)* [4, 10, 23, 30, 31], and (b) *Information Extraction (IE)* [1, 3, 7, 22, 29].

Concerning *SP approaches*, they focus on question understanding, i.e., they convert sentences into their semantic representation and they usually generate a query (e.g., a SPARQL query), for retrieving the answer. Such approaches can answer compositional questions by using aggregation operators (e.g., argmax, count), however, they suffer from structure differences between the Knowledge Base and the input Natural Language question. On the contrary, the objective of *IE approaches* is to identify the main entities of the question and to map the words of the question to the Knowledge base predicates, either by using pre-defined templates, or automatically generated ones. As a final step, these approaches exploit the neighborhood (in the knowledge graph) of each matched entity for producing the final answer. Their disadvantage is that they cannot easily answer compositional questions, since they cannot represent such operators [11]. Our work, i.e., `LODQA`, belongs to IE category.

The most related approaches to `LODQA`, are predominantly *WDAqua* [7] and *AMAL* [22], and secondarily Aqqu [3,32] and SINA [25]. In contrast to these four related tools, `LODQA` exploits the contents of 400 datasets for answering a given question, whereas the other tools support either a single or a few KBs, therefore, they cannot verify the answers from several datasets. `LODQA` follows an *information extraction approach* by exploiting the services and indexes of `LODsyndesis`, instead of using a SPARQL translation approach. By using indexes, we can offer faster question responses comparing to approaches using SPARQL queries, since their efficiency usually rely on the sources' servers, whereas SPARQL querying can be quite expensive for large knowledge bases.

Comparing to WDAqua [7], we take into account both the syntactic form of the question and the relations of the underlying question words, instead of

exploiting only the semantics of the question words. However, we do not support multilingual questions. Concerning the differences with AMAL [22], the latter exploits Wikipedia Disambiguation links and DBpedia lexicons for performing relation matching, whereas we use services offered by `LODsyndesis` for taking into consideration equivalent relationships, and also synonyms through WordNet. Therefore, we can exploit multiple sources for relation matching task, instead of using only DBpedia resources. On the contrary, we do not support list and aggregation questions, which are offered from AMAL [22]. Regarding Aqqu [3], we exploit two different tools for entity detection, i.e., DBpedia Spotlight and Stanford CoreNLP, whereas Aqqu [3] uses hand-crafted rules based on POS-tags. Concerning SINA [25], it performs the data interlinking among the datasets (for a few number of datasets) at query time (which can be time-consuming), whereas, we exploit the indexes of `LODsyndesis`, where the interlinking has already been done once at indexing time. Finally, comparing to Aqqu [3] and SINA [25], we do not use any training data for producing the answer.

3 The `LODQA` Approach

In Section 3.1 we present the `LODsyndesis` services that we exploit, while in Section 3.2, we introduce the proposed QA process.

3.1 LODsyndesis Knowledge Services

We decided to use `LODsyndesis`, for 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. Moreover, it is worth mentioning that (c) it can surpass the problems of non-informative URIs, since it supports cross-dataset reasoning. The process of indexing of `LODsyndesis` is illustrated in Figure 1, i.e., `LODsyndesis` uses as input several datasets containing RDF triples (see the lower left side of Figure 1), where a triple is a statement of the form subject-predicate-object (s,p,o) and T is the set of all the triples that exist in the LOD cloud. Moreover, it uses several equivalence relationships (see the lower right side of Figure 1), such as `owl:sameAs` relationships which denote that two URIs refer to the same entity (e.g., dbp:Heraklion $\equiv$ test:Heraklion), and `owl:equivalentProperty` relationships which are used for denoting that two schema elements are equivalent (e.g., dbp:population $\equiv$ test:population). `LODsyndesis` uses as input these equivalence relationships and computes their transitive and symmetric closure for collecting all the information for any entity (e.g., see the index for "Heraklion" in the middle part of Figure 1).

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 1, one dataset contains a comment about Heraklion, another one about

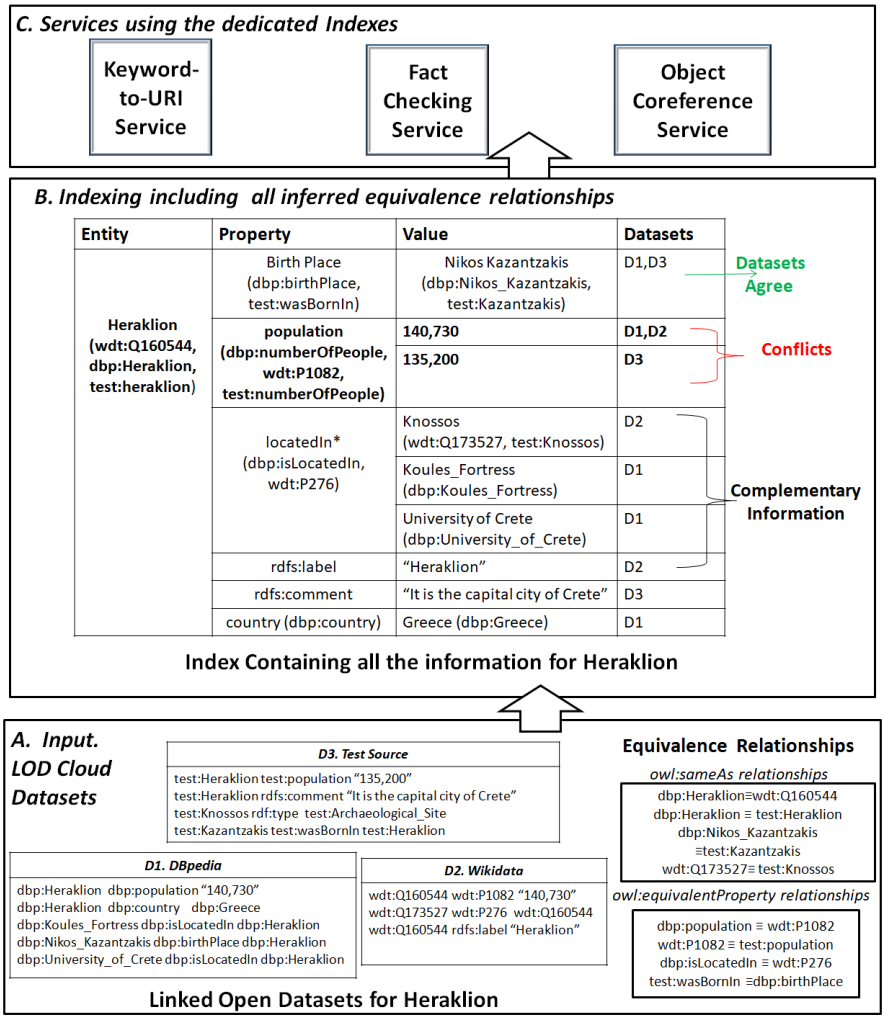

Fig. 1: The steps of `LODsyndesis`

the country where Heraklion is located in, and so forth. Concerning benefit c), many datasets publish non-informative URIs, e.g., in Figure 1 only Wikidata can answer the question "Is Knossos located in Heraklion?", with the corresponding triple (wdt:Q173527,wdt:P276,wdt:Q160544). `LODsyndesis` supports cross-dataset reasoning, i.e., it computes the transitive and symmetric closure of equivalent relationships (e.g., see the lower right side of Figure 1) and it stores the equivalent URIs of each URI, thereby, it knows that dbp:Heraklion ≡ wdt:Q160544, dbp:isLocatedIn ≡ wdt:P276 and wdt:Q173527 ≡ test:Knossos. Therefore, we can find fast the correct answer, by checking the equivalent URIs of each one.

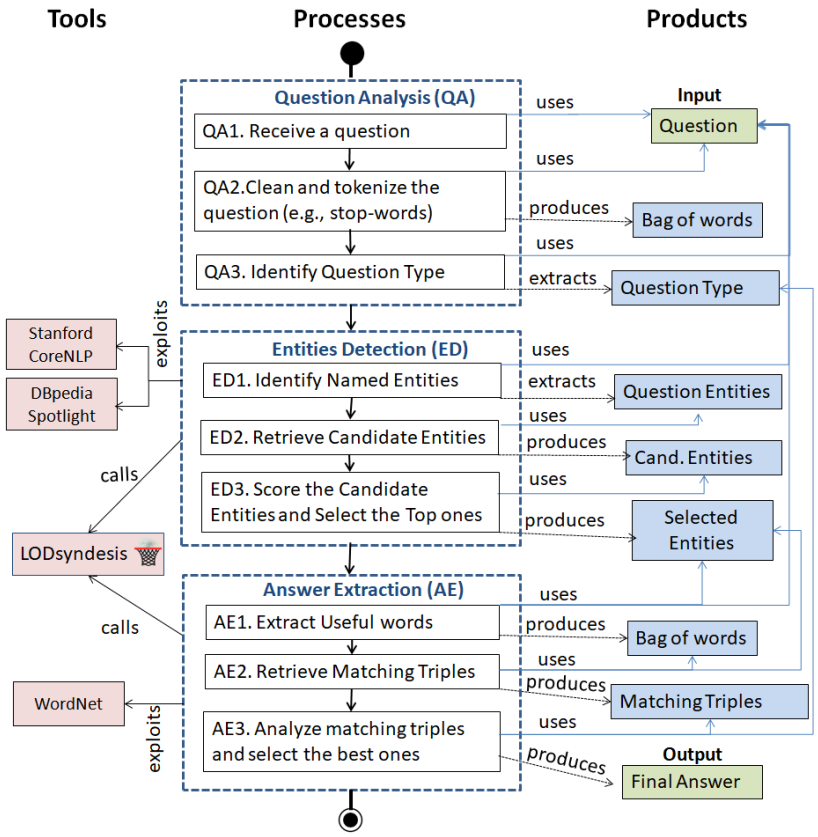

Fig. 2: Overview of the QA Process

`LODsyndesis` offers several services by exploiting the aforementioned semantics-aware indexes. Concerning `LODQA`, it exploits the following three `LODsyndesis` services (more information about them can be found in [18]): (a) the *Keyword-to-URI* service which returns those URIs whose suffix starts with a given keyword, (b) the *Object Coreference* service which provides all the equivalent URIs for a given one, and (c) the *Fact Checking* service, which can be used for retrieving all the triples containing a set of given keywords for a single focused entity.

3.2 The process of `LODQA`

The question answering process consists of three main phases: *(i) Question Analysis (QA)*, *(ii) Entities Detection (ED)* and *(iii) Answer Extraction (AE)*. Figure 2 introduces the main phases and steps of `LODQA` process, where one can clearly observe the input and the output of each different step. To make these steps more clear, Figure 3 shows a running example i.e., the steps for answering the factoid question "What is the population of Heraklion?".

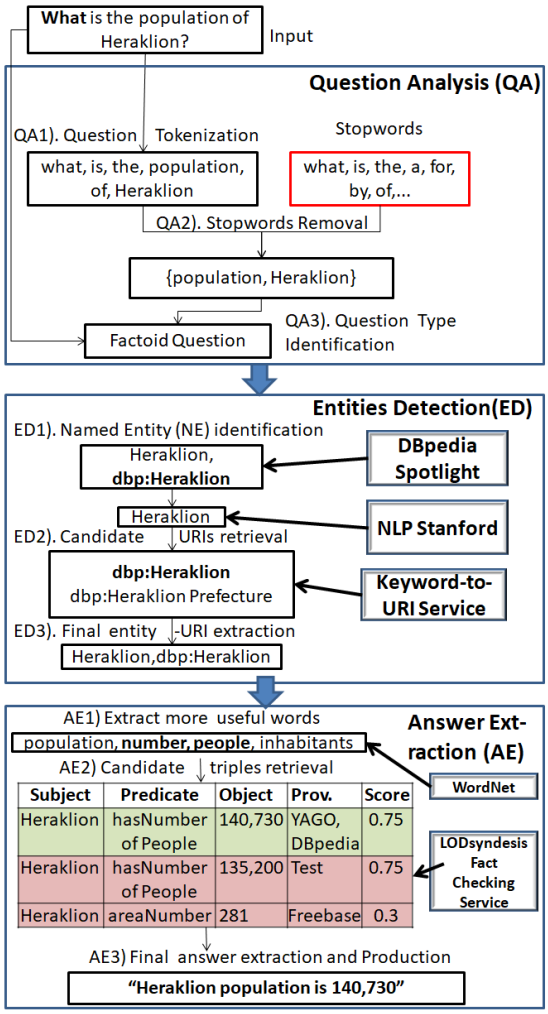

Fig. 3: The QA Process over the running example

***Question Analysis* Phase.** In this phase, we convert the input question into a set of tokens and we remove the stopwords of the given question, such as the words "what" and "is" in the example of Figure 3. The next step is the question type identification, by using a set of indicative words and simple heuristics. Concerning factoid and confirmation questions, we check if the question starts with one of the following words: $W_{factoid} = \{$when, who, where, what, which, ...$\}$ for factoid questions, and $W_{confirm} = \{$are, did, is, was, does, were, do, ...$\}$ for confirmation questions. Finally, for the definition questions, we check if the question contains one of the following words: $W_{def} = \{$mean, meaning, definition, ...$\}$ in its middle part. As an example, in Figure 3 we identified the question as a factoid one, since is starts with the word "what".

Entities Detection **Phase.** The target of the second phase is to identify the entities occurring in the question and to link them with their corresponding URIs in the sources which are supported from `LODQA`. For achieving this goal, we exploit two widely used tools, i.e., Stanford CoreNLP [9, 13] and DBpedia Spotlight [14]. The Stanford CoreNLP tool, hereafter *SCNLP*, combines hand-crafted rules and statistical sequence taggers for identifying the named entities of a given question, and it returns the recognized entities in natural language, e.g., for the input question of Figure3, it will return as the entity of the question the word "Heraklion". However, since `LODQA` needs also the corresponding URI of each entity, we use the *Keyword-to-URI* service of `LODsyndesis`, which returns a set of candidate URIs for a given keyword. On the contrary, DBpedia Spotlight uses a string matching algorithm, a lexicon for retrieving the possible candidates and a TF*IDF variation for tackling disambiguation issues. As an output, it produces for each entity a pair containing the entity in natural language and its corresponding URI in *DBpedia* knowledge base [12]. Afterwards, we compare the candidate URIs derived from both tools for the same entity, we compute a score for each such URI, and we select the most relevant URI for each entity. In our running example of Figure 3, we identified that for the entity "Heraklion", the most relevant URI is "dbp:Heraklion" (and not "dbp:Heraklion_Prefecture).

Answer Extraction **Phase.** In the third phase, the objective is to retrieve the candidate RDF triples, and to identify the best matching triple, for returning the final answer. It is achieved through the exploitation of `LODsyndesis`, and of an *expanded set of question words* by using (i) the SCNLP lemmatizer and (ii) the WordNet dictionary [15]. Concerning SCNLP lemmatizer, we use it for extracting the lemmas of the question words, e.g., the lemma of the word "analyzed" is "analyze". Regarding WordNet dictionary, we use the API offered by extJWNL[1], for deriving nouns, verbs and synonyms based on the POS tags of the question words. Therefore, it can produce from the word "populated" the noun "population", and from the word "population", the word "inhabitants" and the phrase "number of people" which have a similar meaning.

The next step is to exploit the *factChecking* service of `LODsyndesis`, which takes as input a single entity, along with a set of words, i.e., in our running example, we give as input to that service the parameters "dbp:Heraklion" and the words "population", "inhabitants" and "number of people" (e.g., the latter two phrases derived through WordNet dictionary). Afterwards, a set of candidate triples is returned from `LODsyndesis`, which are analyzed through `LODQA` for selecting the most relevant answer for the given question. In our running example, we received three candidate triples for the input question. However, it is worth mentioning that without the *Question Expansion* step, it would be infeasible to derive candidate triples for the given question (i.e., there was not a triple containing the word "population" in this example).

Afterwards, `LODQA` produces the final triple for the input question, by taking into account its type. Specifically, for factoid questions, it selects the max

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

scored triple based on the percentage of the question words included in the triple, and the number of provenance datasets. In our running example, by analyzing the candidate triples, we identified that the first two triples were more relevant comparing to the third one. However, `LODQA` selected the first triple (i.e., the population of "Heraklion" is "140,730") as the best matching one, since it was included in more datasets in comparison to the second triple. Concerning confirmation questions, `LODQA` returns "Yes" if the candidate triple contains all the entities of the question and at least one other "useful" word, e.g., suppose that the given question is the following: "Was Nikos Kazantzakis born in Heraklion?" and the candidate triple is "Nikos Kazantzakis, birthPlace, Heraklion", the system would return "Yes", since the answer contains both entities and the predicate birthPlace, which is a synonym to the predicate "born in". Finally, for definition questions, it returns the best matching triple containing as predicate one of the following: *rdfs:comment, dcterms:description, dbpedia:abstract.*

4 Evaluation

Here, in Sections 4.1-4.4 we report experimental results concerning the effectiveness and efficiency of *Entities Detection* and *Answer Extraction* steps, by using SimpleQuestions (v2) collection [6] [2]. Finally, in Section 4.5, we show some measurements regrading the impact of using multiple datasets.

4.1 Evaluation Collection

We performed a comparative evaluation over the SimpleQuestions(v2) collection [6], for evaluating and improving the tasks of *Entities Detection* and *Answer Extraction*. This collection contains 108,442 simple (mainly factoid) questions, i.e., questions that can be answered by using a single triple from Freebase knowledge base [5]. For each question it includes the corresponding answer, i.e., a single Freebase triple. It is worth mentioning that `LODsyndesis` contains information from several sources (including DBpedia, Freebase and others), therefore `LODQA` can answer a question by exploiting a different dataset (e.g., DBpedia) and not Freebase. Since this requires a manual check for evaluating whether the answer is correct, mainly due to missing mappings between these sources (e.g., between DBpedia and Freebase), we selected a subset of them for the experiments. Indeed, we selected randomly a set 1,000 factoid questions, where each question contained on average 7 words. The subset of the collection that was used in the experiments, is accessible online [3].

4.2 Entities Detection Evaluation

Our objective is to understand how the capabilities of the two used different tools (SCNLP and DBpedia Spotlight) affect the outcome of the whole process. For

[2] `http://research.fb.com/downloads/babi/`
[3] `http://islcatalog.ics.forth.gr/tr/dataset/simplequestions-v2-1000-questions`

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

Model	Accuracy	Accuracy (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-LNV	0.407	0.547

Table 1: Evaluation using 1000 questions from SimpleQuestions(v2). Left: Accuracy of each *Named Entity Recognition* approach. Right: Accuracy of each *Triples Retrieval approach*

this reason, we report comparative results by using three different approaches. Specifically, for each approach we measure the accuracy, i.e., the number of questions where each approach identified the correct entities, divided by the number of all questions. The approaches which are compared for the Named Entity (NE) detection and linking follow: (i) *SCNLP-Spotlight*: we use SCNLP and in case of failing to recognize any NE, we use DBpedia Spotlight, (ii) *Spotlight-SCNLP*: we use DBpedia Spotlight and in case of failure, we use SCNLP, (iii) *Combined*: we exploit both tools for identifying the NEs and their URIs and then, we use some simple heuristics for selecting the best entities. The evaluation results are shown in Table 1(left). We observe that the combined method achieves much higher accuracy, i.e. 0.73, compared to any of the other two approaches, which achieve an accuracy of 0.62 and 0.65, respectively! For this reason, we will use that method for evaluating the outcome of the whole process in Section 4.3.

4.3 Answer Extraction Evaluation

Regarding the *Answer Extraction* step, our target is to tackle the possible *lexical gap* between the question and the underlying datasets. For this reason, we compare variations of our approach, where we expand the available set of questions words. Indeed, we perform the expansion by producing the lemmas (from SCNLP) of the question words. Moreover, based on the POS tag of each word, if (a) a word is a *Verb*, we produce all the derived nouns (from WordNet) and (b) if it is a *Noun*, we produce all the derived verbs. We evaluate the effectiveness of our approach (i.e., LODQA) by using all the aforementioned expansion methods (i.e., lemmas, nouns, verbs), and we compare it with variations of our approach that do not perform word expansion based on lemmas (LODQA-w/o-L), nouns (LODQA-w/o-N), and verbs (LODQAw/o-V). Moreover, we provide also experimental results for an approach that do not perform any word expansion, i.e., (LODQA-w/o-LNV). The evaluation results are shown in Table 1(Right), where we measure the accuracy of each different variation, i.e., the number of questions answered correctly, divided by the number of all the questions. The proposed approach (i.e., LODQA) using all the expansion steps achieves the highest accuracy (i.e., 0.49), whereas by taking into account only the questions that passes the Entities Detection Step (i.e., all the questions that we detected the correct enti-

Step	Average Time
Question Analysis	0.007 s
Entities Detection	1.808 s
Query Expansion	0.330 s
Candidate Triples retrieval	3.005 s
Final Answer Production	0.134 s
Total Time	5.330 s

Table 2: Efficiency Results using 500 questions from SimpleQuestions(v2).

Measurement	Value
Number of Entities in $\geq$ 2 datasets	25,289,605
Number of Entities in $\geq$ 3 datasets	6,979,109
Verifiable questions from at least 2 datasets	28,439,760
Average triple per entity (by using 1 dataset)	17.3
Average triple per entity (by using all datasets)	29.3

Table 3: `LODsyndesis` Measurements

ties), the accuracy increases (i.e., 0.64). On the contrary, without any question words expansion, the precision is only 0.4 and 0.54 respectively. Concerning the different methods for expanding the set of question words, we identified that for this set of questions, verbs were more important that nouns and lemmas. The above evaluation results indicate that our approach is KB agnostic, since it can be applied for any given KB (indexed by `LODsyndesis`) without requiring any additional effort and training data.

4.4 Efficiency

For performing the experiments, we used a single machine with 8GB RAM, 8 cores and 60GB Disk space, and we measured the efficiency in 500 questions of SimpleQuestions collection for each different step, as it can be seen in Table 2. As we can see, `LODQA` needs on average 5.33 seconds to answer a question. The most time consuming steps are to retrieve the candidate triples (57.2% of the required time) and to detect the entities of the question (30% of the required time). Furthermore, it is worth noting that the minimum time for answering a question was 1.6 seconds and the maximum one was 37.46. Finally, half of the questions (i.e., median value) were answered in less than 3.7 seconds.

4.5 The Benefits of using Multiple Datasets

Table 3 shows measurements for evaluating the impact of using multiple datasets (and of performing cross-dataset reasoning). Particularly, `LODsyndesis` contains information for 25.2 million entities from at least two datasets, whereas for 6.9 million entities we can retrieve information from at least three datasets. It is worth noting that there exists 28.4 million of possible questions that can be verified by more than one dataset, i.e., corresponds to simple questions answerable from at least two datasets. Therefore, it is evident that by using multiple datasets, we increase the probability of answering a given question, whereas the

number of verifiable questions is increased, too. Moreover, for these 25.2 million entities, if we use only a single dataset (even the dataset containing the most triples for each entity), the average number of triples per entity is 17.3. On the contrary, due to the cross-dataset reasoning (i.e., computation of transitive and symmetric closure of equivalent relationships), `LODsyndesis` collects all the available information for each entity from all the datasets. Due to this process, the average number of triples of each of these entities highly increases (i.e., it becomes 29.3).

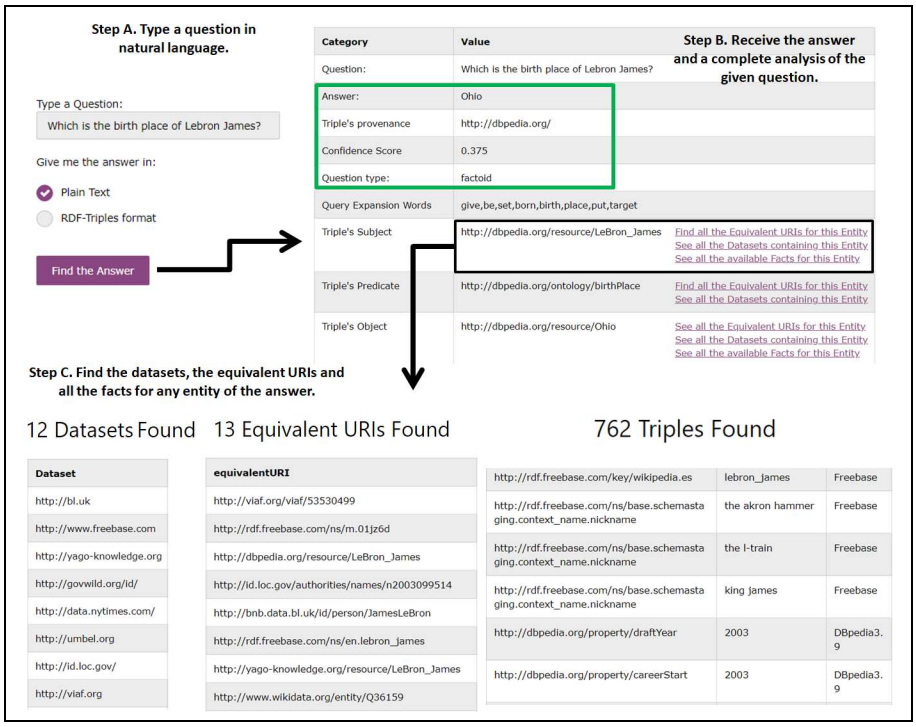

Fig. 4: An example of the LODQA demo

5 Web Demo and Related Links

The `LODQA` is currently hosted and runs in a single machine of *okeanos* cloud computing service (`https://okeanos.grnet.gr/`) with an *i5* core, 8 GB main memory and 60 GB disk space. Although the hosting machine has a low computational power, the interaction is real time, i.e., few seconds are needed to answer a question.

In the *website* `http://83.212.101.193:8080/LODQA/DemoQuestions`, we offer a list of demo factoid, confirmation and definition questions, enabling the user

to run questions for each of these categories. Three indicative question-answer pairs, one for each question type, are in order: (Which was the birth place of Socrates?, Athens), (Is Nintendo located in Kyoto?, Yes!), (What is Parthenon?, the parthenon, a temple built in honor of athena...).

As we can see in Figure 4, for the question "Which is the birth place of Lebron James?", `LODQA` returns the answer (i.e., "Ohio" in this example), and also a complete analysis for the question (see the right part of Figure 4). Specifically, except for the short answer, one can find more information about its *provenance*, its *type* and its *confidence score*. Moreover, it returns the triple (in RDF format) where we found the question, whereas one can explore more information for each entity which is part of the triple. For example, as it is shown in the lower side of Figure 4, one can explore for the entity "Lebron James" all its URIs (i.e., 13 URIs in total), the datasets where this entity occurs (i.e., 14 datasets) and one can have access to all the facts (i.e., 762 triples) about that person! Finally, a *tutorial video* is accessible in `https://youtu.be/bSbKLlQBukk`, whereas an *online demo* that will allow any user to ask questions will be released soon.

6 Conclusion

`LODQA` is a Question Answering approach that exploits hundreds of Linked Data sources, instead of using a single or few KBs (such as in [7, 22, 25, 32]). By using multiple datasets, the number of answerable questions increases, whereas the validity of any answer can be estimated from several sources, and answers are shown along with their provenance. We introduced an approach that exploit hundreds of datasets simultaneously and follows a variety of methods, without requiring any training data, for answering a question expressed in natural language. In particular, it includes methods for question cleaning, heuristic based question type identification, entity recognition, linking and disambiguation using Linked Data-based methods and pure NLP methods (specifically DBpedia Spotlight and Stanford CoreNLP), WordNet-based question expansion for tackling the lexical gap (between the input question and the underlying sources), and triple scoring for producing the final answer.

Concerning the evaluation, we used 1,000 questions of SimpleQuestions (v2) collection [6]. The evaluation show that regarding Entities Detection step, the combined method that exploits both entity recognition tools achieves the highest accuracy. This reflects the importance of using both KB-agnostic tools (SCNLP) and Large-scale KB-based tools (DBpedia Spotlight) for the tasks of entities recognition, linking and disambiguation. Regarding Answer Extraction, it seems that by using all the word expansion steps, we achieve the highest accuracy, while the method which does not perform any question words expansion (does not consider the lexical gap) achieves the worst results. This evidences the importance of tackling the lexical gap between the input question and the underlying sources for retrieving relevant information, i.e., triples. Moreover, we introduced experiments about the efficiency of our approach (e.g., half of the questions can be answered in less than 3.7 seconds) and the benefits of this approach in terms

of answerable questions and answer verification (e.g., 28.4 million questions can be verified by at least two datasets). As a future work, we plan to improve the system for making it capable of returning responses to more complex questions, i.e., list questions or questions that require combining paths of triples.

Acknowledgements. The research work was supported by the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under the HFRI PhD Fellowship grant (GA. No. 166).

References

1. A. Abujabal, M. Yahya, M. Riedewald, and G. 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. K. Affolter, K. Stockinger, and A. Bernstein. A Comparative Survey of Recent Natural Language Interfaces for Databases. *arXiv preprint arXiv:1906.08990*, 2019.
3. 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*, pages 1431–1440. ACM, 2015.
4. J. Berant and P. Liang. Imitation learning of agenda-based semantic parsers. *Transactions of the Association for Computational Linguistics*, 3:545–558, 2015.
5. K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM, 2008.
6. A. Bordes, N. Usunier, S. Chopra, and J. Weston. Large-scale Simple Question Answering with Memory Networks. *CoRR*, abs/1506.02075, 2015.
7. D. Diefenbach, K. Singh, and P. 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.
8. E. Dimitrakis, K. Sgontzos, and Y. Tzitzikas. A Survey on Question Answering Systems over Linked Data and Documents. *Journal of Intelligent Information Systems*, pages 1–27, 2019.
9. J. R. Finkel, T. Grenager, and C. 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.
10. S. Hakimov, S. Jebbara, and P. Cimiano. AMUSE: multilingual semantic parsing for question answering over linked data. In *International Semantic Web Conference*, pages 329–346. Springer, 2017.
11. K. Höffner, S. Walter, E. Marx, R. Usbeck, J. Lehmann, and A.-C. Ngonga Ngomo. Survey on challenges of question answering in the semantic web. *Semantic Web*, 8(6):895–920, 2017.
12. J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195, 2015.

13. C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.
14. P. N. Mendes, M. Jakob, A. García-Silva, and C. 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.
15. G. A. Miller. WordNet: A Lexical Database for English. *Commun. ACM*, 38(11):39–41, Nov. 1995.
16. A. Mishra and S. K. Jain. A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, 28(3):345–361, 2016.
17. M. Mountantonakis and Y. Tzitzikas. High Performance Methods for Linked Open Data Connectivity Analytics. *Information*, 9(6):134, 2018.
18. M. Mountantonakis and Y. Tzitzikas. LODsyndesis: Global Scale Knowledge Services. *Heritage*, 1(2):335–348, 2018.
19. M. Mountantonakis and Y. Tzitzikas. Large Scale Semantic Integration of Linked Data: A survey. *ACM Computing Surveys (CSUR)*, 52(5):103, 2019.
20. A. Papangelis, P. Papadakos, Y. Stylianou, and Y. Tzitzikas. Spoken Dialogue for Information Navigation. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 229–234, 2018.
21. B. Patra. A survey of Community Question Answering. *CoRR*, abs/1705.04009, 2017.
22. N. Radoev, M. Tremblay, M. Gagnon, and A. 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.
23. S. Reddy, O. Täckström, M. Collins, T. Kwiatkowski, D. Das, M. Steedman, and M. Lapata. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140, 2016.
24. A. Rodrigo and A. Peñas. A study about the future evaluation of Question-Answering systems. *Knowledge-Based Systems*, 137:83–93, 2017.
25. S. Shekarpour, E. Marx, A.-C. N. Ngomo, and S. 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.
26. K. Stockinger. The Rise of Natural Language Interfaces to Databases. 2019. ACM SIGMOD Blog, June 2019.
27. Y. Tzitzikas, N. Manolis, and P. Papadakos. Faceted exploration of RDF/S datasets: a survey. *Journal of Intelligent Information Systems*, pages 1–36, 2016.
28. M. Wang. A survey of answer extraction techniques in factoid question answering. *Computational Linguistics*, 1(1), 2006.
29. X. Yao, J. Berant, and B. Van Durme. Freebase QA: Information extraction or semantic parsing. In *Proceedings of ACL*, 2014.
30. S. Yavuz, I. Gur, Y. Su, M. Srivatsa, and X. 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.
31. W.-t. Yih, M.-W. Chang, X. He, and J. Gao. Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base. In *Proceedings of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, volume 1, pages 1321–1331, 2015.
32. Y. Zhang, S. He, K. Liu, and J. Zhao. A Joint Model for Question Answering over Multiple Knowledge Bases. In *AAAI*, pages 3094–3100, 2016.