Question Answering over Knowledge Bases

Lucia Siciliani(✉)

Department of Computer Science, University of Bari Aldo Moro, Bari, Italy
lucia.siciliani@uniba.it

Abstract. The fast growth of the Semantic Web has unleashed its potentialities, leading to the development of many tools and services that can exploit the huge amount of information it contains. As more semantic information is available online, mainly in the form of ontology based Knowledge Bases the process of searching and querying this content has become more and more challenging. Question Answering, which defines the task of retrieving an answer to a question formulated using natural language, can make the Semantic Web easily accessible by anyone, even by those users that do not know how to use a specific data query language or are unaware of the structure of the KB they want to query. Moreover, in the same way the Semantic Web can benefit in its spread from Question Answering, also Question Answering systems can improve their outcome since Knowledge Bases can be exploited to retrieve a concise answer for complex questions. Although several approaches have been proposed by the research community in the field, the experimental results show that the performance are still far from optimal. Following the future directions presented in the latest works about this field, we outline an approach for Question Answering over structured data applicable to Knowledge Bases whose aim is to overcome the main issues that affect the research in this area.

Keywords: Question Answering · Semantic Web
Natural language processing · Machine learning · Knowledge Bases

1 Introduction

Question Answering (QA) has the purpose of retrieving an answer to a question written in natural language by a user. To a certain extent QA systems can be considered as a particular kind of Information Retrieval (IR) systems, since in both the cases the goal is to find an answer to a specific information need expressed in textual form. The main difference between them regards the kind of output: in an IR system the output is usually represented by a subset of the overall corpus of documents, ordered according to a criterion of relevance, meanwhile in QA systems the output can take the form of a concise fact, in the case of factoid QA, or other kind of formats such as lists, short passages, and so on.

Among several different classifications for Question Answering, one of the most discriminating concerns the type of data sources. According to such taxonomy, the data stored into the resource exploited by the QA system can be either unstructured or structured. Unstructured data, as the name suggest, takes the form of free text without any kind of arrangement. The source text can be preprocessed using a wide range of techniques of various complexity, in order to facilitate the retrieval of the query keywords in the following steps. On the other hand, structured data is already organized in data structures which allow the system to leverage them almost directly. Obviously, many combination are possible and one opportunity is also represnted by web services, from which both structured and unstructured data can be obtained and exploited by a question answering system by applying different kind of techniques.

QA over structured data is rooted in the late sixties and early seventies, when the first Natural Language Interfaces were developed as a way to access data contained into databases [1]. Later on, the attention focused over the extraction of relevant information from free text. In the last two decades, the birth of the Semantic Web (SW) [2] has opened up new paths for QA.

SW was born as an extension of the so called Web 2.0 with the aim of enriching the huge amount of data already available online with a meaning. Thanks to SW an increasing amount of information is accessible on the Web in the form of Knowledge Bases, such as DBpedia [3], YAGO [4] and Wikidata [5], which collect huge amounts of information using the ontology model. Nowadays all these projects are still ongoing, with a steady effort to further extend and improve the quality of data that they include. A crucial aspect to consider in order to exploit such amount of information is how to actually access it and this is where QA systems prove to be useful.

Moreover QA systems over structured data can benefit in several ways by the use of KBs: first of all, they can allow the system to enhance the answer with contextual informations (e.g.: images, infoboxes), secondly, it is possible to aswer more complex questions by combining information coming from different KBs.

The format chosen to model data in the SW is the Resource Description Framework (RDF) and the query language associated to it is the SPARQL Protocol and RDF Query Language. Although those models have been properly designed to guarantee the automated processing of Web resources and the interoperability among applications that use this kind of information, they still represent an obstacle for many users who are not accustomed to such formalisms. Thus, QA systems can be used to overcome such limits and disclose to a greater variety of users the information encoded into KBs.

2 State of the Art

The problem of QA over KBs has been addressed in a wide variety of ways. Until the last decade there was still no official benchmark for evaluating those systems, hence a comparison was almost infeasible. However, the growing interest towards the potentialities of the SW has raised the need to guide the efforts made in this research field and try to find a solution for this problem. At the moment there are three major benchmarks for QA system over KBs, namely: QALD [6–11], WebQuestions [12] and SimpleQuestions [13]. Further details about those datasets will be provided in Sect. 6.

QA systems share the common feature to be very complex since transforming a question in natural language to its equivalent in SPARQL is not a straightforward task.

Such conversion must start from the analysis of the user question via different techniques, whose aim is to catch the overall meaning of the question and collect information which facilitate the following phases. The methods than can be adopted include Part-Of-Speech tagging, parsing, the identification of the kind of question or of the expected answer type. Next, the phrases contained in the questions must be mapped with the entities or relations belonging to RDF datasets. Selecting the right resource is crucial since even a slight error could greatly affect the meaning of the final query that could return a wrong answer.

Each of these phases can obviously be further decomposed in smaller subtasks and all of them can be performed using a plethora of methods. For this reason is really hard to categorize the approaches proposed in the literature, even though there are some surveys that accomplished this difficult goal.

In [14], the authors give an overview of the topic describing the actual state of the research, categorizing the main approaches proposed in the literature, giving information about the available tools which can be used to design a novel system and providing a description of the evaluation metrics along with the available benchmarks.

In [15], the authors tackle the problem from an interesting perspective, focusing the attention over the main challenges related to this field and describing each one of the selected publications according to them.

One of the latest survey [16], provides an accurate analysis of the state of the art, listing almost all the systems proposed for every benchmark. After having traced the overall structure of a QA system, each component is further analyzed to describe the different approaches that have been employed and which systems have used them.

Since in this context an exhaustive description of all the work proposed until now is infeasible due to space constraints, we will focus the attention on the systems that achieved the most significant results in QALD (starting from its fourth edition), SimpleQuestions and WebQuestions.

Xser [17] is the system that obtained the best performance in both QALD-4 and QALD-5. To retrieve the answer for a given question it utilizes two different stages: the first one which is KB-independent and the second one which is KB-dependent. In the first phase, the user query is subdivided into phrases and to each of them a semantic label (i.e. entity, relation, category, variable) is assigned. Those labels are then used to construct a Directed Acyclic Graph (DAG) which encodes the query intention. The main advantage of performing those steps independently from the KB is that the method can be more easily adapted to any new KB. The KB-dependent phase has the aim of instantiating the query intention with regard to the chosen KB. For each semantic label, different tools are used to create a list of possible candidates in the KB and to disambiguate among them. Once this step is completed, the query can finally be translated into its structured equivalent form. The main disadvantage of this approach is that the semantic parser used to produce the DAG must be trained with a corpus of questions that must be manually annotated with the corresponding labels.

CANaLI [18] achieved an outstanding result during the sixth edition of QALD, proposing an approach based on controlled natural languages. A controlled natural language can be seen as a restricted version of a certain natural language, obtained by considering only a subset of its vocabulary and its grammar rules. The analysis of the question is then performed by exploiting a finite state automata. CANaLI progressively considers each phrase in the question and, if it belongs to those accepted by the system, it shifts the state of the automata accordingly to the chunk in input. Once all the question is analyzed, the history of all the states that have been reached by the automata along with the transition rules that have been applied are used to construct the SPARQL query. Since this process is deterministic, the query generation step can be performed without mistakes. However this approach reveals some drawbacks: first of all, if a certain phrase is not recognized by the automata then the final state is never reached and the computation can not go on; secondly the question must be formulated using the grammar rules which are accepted by the system. This is why the authors performed a rephrasing of the questions that belonged to the QALD, adapting the vocabulary and the syntactic structure when possible.

In QALD-7 [11] the task of English QA over Wikipedia was introduced for the first time along with the one of multilingual QA over DBpedia which appeared in every edition of the challenge.

The best performances in the task using DBpedia as RDF dataset were obtained by ganswer2 [19]. The authors propose a solution where the answer to the user question is obtained without generating the SPARQL query. The natural language query is interpreted as a semantic query graph and then the system tries to match such structure in the overall RDF graph of the KB. In this way it is also possible to postpone the disambiguation of possible candidates for the matching, thus decreasing the response time.

Regarding the task focused on Wikidata instead, WDAqua [20] managed to obtain a better result compared with the other participants. The system is language and KB independent, meaning that it can answer to a question formulated in different languages and can exploit different KBs. In order to achieve this result, the authors start from the assumption that what really encloses the meaning of a question is the semantic of the words that compose it rather than its syntactic features. Following this assumption, all the candidate entities, properties and classes which can be extracted by analyzing the source question are extracted, while stopwords are removed. Next, following different patterns, a set SPARQL queries is constructed and each query is then ranked according its capability of covering the meaning of the original question. Finally, using a model based on logistic regression, the system estimates the confidence in the retrieved answers. If such value is greater than a certain threshold, the answer is given, otherwise it is rejected.

Regarding SimpleQuestions and WebQuestions, the system that performed the best on these datasets are [21] and [22], respectively.

SimpleQuestions contains a big amount of questions, so most of the systems that have been evaluated on this dataset use an approach based on neural networks. [21] makes no exception, making use of a two step approach. Since all the questions only involve a single binary relation, the first step consists in identifying the phrases which can correspond to the subject and the predicate of the question. Next, entity linking and relation detection are performed to actually map those phrases to entities and relations in the KB. Those steps are not sequential, but they are handled together using a Recurrent Neural Network.

[22] use a Memory Network to answer questions over WebQuestions. First all the possible n-grams of words in the question within the dataset are matched against the entities in the KB. This list of matching items is used to construct a set of hypothetical questions that are compared to the upcoming question using a similarity measure, which is then exploited to find out the correct answer.

3 Problem Statement and Contributions

As previously stated in Sect. 1, the growth of the Semantic Web has raised the problem of how effectively exploit this huge amount of information.

One of the main problems which prevents the spread of this technology among common users is represented by the difficulty of extracting the information contained in a KB. In fact in order to do this, it is necessary to know the vocabulary adopted in the KB and use a specific Data Query Language (i.e. SPARQL) that can be challenging to use, especially when trying to satisfy complex information needs.

QA systems aim to free users from these constraints allowing them to query a KB using natural language. In most cases, the question is formulated using a terminology which is greatly different from the vocabulary of the knowledge base since the user is completely unaware of it. For example, given the question *"Who is the writer of Neuromancer?"* and DBpedia as the only KB to be queried, it is not possible to find an answer if only the label of each resource is considered as valid. Infact, the entity *Neuromancer* has no property labeled *writer* and the right term that should be used in accordance to the structure of DBpedia is *author*. This issue is known in literature with as lexical gap. Finding an effective way to bridge this gap is the primary problem that we want to deal with.

The problem represented by the lexical gap is intertwined to another important issue: ambiguity. Ambiguity can be considered as an intrinsic feature of natural language, where the meaning of a word is deeply influenced by the context in which it occurs. Considered by itself, ambiguity is not a negative characteristic: it adds syntactic sugar to a language and is the fundamental element to all the figures of speech which enrich our cultural heritage. However, going back to the context of QA over KBs, ambiguity is something that we want to avoid, since it can lead to an incorrect answer. In the question *"What is the genre of Chicago?"* the entity *Chicago* could refer to the city, the band or the musical, however the word *genre* can give a good hint about which option is the correct one. A QA system has to cope with polysemous words performing a disambiguation step, which obviously must be executed after entity linking and relation prediction that allow to collect a set of candidate resources related to the phrases within the question.

Even if the lexical gap and the ambiguity issues appear to be more focused on the semantics of the user question, its syntactic structure must not be underrated. This is particularly true when handling articulated questions. SPARQL is a powerful query language, which allows to express even complex constructs using a wide range of query forms (e.g.: SELECT, ASK), patterns (e.g.: ORDER BY, DISTINCT) and modifiers (e.g.: GROUP BY, HAVING, FILTER, OPTIONAL). The main problem is that with natural language these constructs can be formulated using a considerable number of expressions that share the same meaning but can have very little in common from a lexical point of view. The goal then is to identify those expressions in the question and properly translate them using SPARQL.

Therefore, the aforementioned issues can be summarized through the following questions:

RQ1 What methods can help bridging the gap between the vocabulary of the user and the one used within a KB? Which one is most suitable method for entities and which one for relations?
RQ2 How to disambiguate among different resources that can be connected to the same phrase in the question?
RQ3 How to understand the overall meaning of the user query and translate it into a proper SPARQL query?

4 Research Methodology and Approach

With our work we want to further extend the results already put forward in literature, trying to face the challenges that are still open in this field.

We can split the architecture of the QA system to be proposed in two distinct pipelines: one for data preprocessing and the other one for the answer retrieval.

The information contained in the KB is essential in the last step of those QA systems that first perform the translation of the user question in a SPARQL query which is then ready to be submitted to the underlying data resource.

Nevertheless this data play a key role in the answer retrieval pipeline, especially when trying to perform entity linking and relation prediction. For this reason it is really important to perform a data preprocessing step.

First of all, the RDF triples of the KB must be collected and indexed in proper data structures. In this way they can be retrieved more quickly when needed, leading to an improvement of the performance of the system.

Secondly, we want to add more information to those structure, by performing a proper augmentation of the KB. In order to do this we could exploit the mappings between questions and query that can be found in the datasets available in literature and even employ some external corpora of text. From this point of view our research could benefit from a focused analysis of the state of the art systems in the fields of entity linking and relation prediction.

The second pipeline is related to answer retrieval, i.e. performing the steps needed to actually find an answer for the question issued by a user.

Given a natural language question we must perform a syntactic and semantic analysis which consists of Part of Speech tagging, semantic parsing and Named Entity Recognition among the others. In literature there are several tools that can be used to perform this analysis, one of them being Stanford Core NLP [23].

Next, we have to map the phrases contained in the question to resources within the knowledge base and disambiguate between them if more than one option is available. This step will be deeply influenced by the preprocessing one, however we plan to make use of word embeddings and other distributional semantic models.

Lastly, the question must be converted into a SPARQL query. We intend to explore the benefits that would come from the introduction of an hybrid approach.

In literature this step has been performed in several ways, but the interesting fact is that, to the best of our knowledge, every system adopts the same mechanism for any kind of query.

We want to combine different approaches in order to mitigate the shortcomings that would descend from the use of a unique method. Usually simpler questions share a similar structure and the differences among them concern only the subject and the relation being involved. An approach based on pre-defined templates to be instantiated in real time has proven to be effective, but it is not as adequate when handling complex queries. For articulated questions, we could exploit as much as possible the output of the semantic analysis using a set of compositional rules.

The combination of these two techniques will allow our system to answer a broader variety of questions, accordingly to their complexity.

5 Preliminary Results

We have already developed a QA system over KBs enhancing the work presented in CANaLI [18].

As described in Sect. 2, an approach using controlled natural language suffers of some drawbacks connected to the constraints imposed on the vocabulary and the applicable grammar rules.

In particular, the vocabulary accepted by the finite state automata used in CANaLI is created by collecting the labels of the resources in the KB and hence only those labels can be employed in the question. For this reason we have decided to extend this vocabulary using Word2Vec (W2V) [24]. In the offline phase, we applied W2V to Wikipedia abstracts to create a word embedding for each term. Then, in the online phase, when a phrase is not recognised by the automata, it is projected in the same distributional space created from the Wikipedia abstract to retrieve all the semantically similar words. The top ten ranked words are selected as suitable candidates and the system substitutes the original word with candidate until the right one is found.

Another issue that we tried to solve is connected to the transition mechanism of the automata, which stops the computation for misinterpreted tokens. To avoid this problem we strengthened the automata with a backtracking algorithm which allows to cancel the last state shift and choose another interpretation the previous phrase.

Although these changes led to an improvement of the system, they could not completely overcome the limits imposed by the adoption of a controlled natural language. When trying to test the result of our version of the system over the QALD-6 dataset, we could avoid the rephrasing of several questions, but still an evaluation on the questions as they are provided by the proposer of the challenge is infeasible and did not produce promising results. Indeed, as shown in [18], this kind of systems prove to be really effective in real use cases when combined with a query autocompletion module, able to guide the user in formulating a query compliant with the controlled natural language, otherwise it requires a manual rephrasing of the question.

Since we want to evaluate our system on benchmarks like QALD, SimpleQuestion and so on, we want to avoid the rephrasing step which could be infeasible for large datasets and could not guarantee a fair comparison with other QA systems. For these reasons, we decided to investigate other approaches.

6 Evaluation Plan

As introduced in Sect. 2 the three main benchmarks for QA over KBs are QALD, SimpleQuestions and WebQuestions. QALD is actually a series of evaluation campaigns for Question Answering over Linked Data which first started in 2011. Every year a new edition of the challenge has been proposed with more enlarged datasets and new tasks. The main RDF dataset used in the QALD challenges is DBpedia even if in the current edition there is a specific task focused on Wikidata. The question comprised in QALD are manually created by the organizers for each edition and they include questions of varying difficulty that can range from simple ones, involving only one binary relation, to complex, which require to handle specific modifiers.

On the other hand, SimpleQuestions and WebQuestions make use of Freebase [25] which has currently been discontinued and is part of Wikidata. These two dataset contain more questions than those in QALD since they are created through a crowd-sourcing method. SimpleQuestions owes its name to the structure of its questions that usually follow a similar pattern and does not involve any kind of reification, unlike what happens for WebQuestions.

Considering the features of these three datasets, the idea is to make an initial evaluation starting from SimpleQuestion: in this way we might exploit the rich dataset of questions and focus the attention on the resolution of the lexical gap and the ambiguities. In a second phase, we can shift our attention over articulated queries involving more than a single relation, testing our system over QALD and WebQuestions.

For the comparison with other systems we will use the standard measures adopted in this field, i.e.: error count, precision, recall and f-measure.

7 Conclusions

In this work we provide an overview of QA over KBs, ranging from a description of the most relevant systems that have been proposed until now, to an analysis of the main issues related to this topic.

Being in an early stage, our work is still focused on the analysis of the approaches proposed by the state of the art system.

Based on our current knowledge, we outline an architecture whose aim is to resolve three principal problems: bridging the lexical gap, resolving the ambiguities and handling complex queries. We expect to achieve good performance by making use of external corpora to facilitate the phrase mapping step and adapting the methods used to retrieve an answer accordingly to the question complexity.

The future step of our research program is to convert this architecture into an actual prototype with the aim of addressing the aforementioned issues in an incremental fashion.

References

1. Androutsopoulos, I., Ritchie, G.D., Thanisch, P.: Natural language interfaces to databases-an introduction. Nat. Lang. Eng. **1**(1), 29–81 (1995)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Sci. Am. **284**(5), 34–43 (2001)
3. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
4. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, pp. 697–706. ACM (2007)
5. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
6. Lopez, V., Unger, C., Cimiano, P., Motta, E.: Evaluating question answering over linked data. Web Seman. Sci. Serv. Agents World Wide Web **21**, 3–13 (2013)
7. Cimiano, P., Lopez, V., Unger, C., Cabrio, E., Ngonga Ngomo, A.-C., Walter, S.: Multilingual question answering over linked data (QALD-3): lab overview. In: Forner, P., Müller, H., Paredes, R., Rosso, P., Stein, B. (eds.) CLEF 2013. LNCS, vol. 8138, pp. 321–332. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40802-1_30

8. Unger, C., Forascu, C., Lopez, V., Ngomo, A.C.N., Cabrio, E., Cimiano, P., Walter, S.: Question answering over linked data (QALD-4). In: Working Notes for CLEF 2014 Conference (2014)
9. Unger, C., Forascu, C., Lopez, V., Ngomo, A.C.N., Cabrio, E., Cimiano, P., Walter, S.: Question answering over linked data (QALD-5). In: Cappellato, L., Ferro, N., Jones, G.J.F., SanJuan, E. (eds.) CLEF (Working Notes), vol. 1391, CEUR Workshop Proceedings, CEUR-WS.org (2015)
10. Unger, C., Ngomo, A.-C.N., Cabrio, E.: 6th open challenge on question answering over linked data (QALD-6). In: Sack, H., Dietze, S., Tordai, A., Lange, C. (eds.) SemWebEval 2016. CCIS, vol. 641, pp. 171–177. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46565-4_13
11. Usbeck, R., Ngomo, A.-C.N., Haarmann, B., Krithara, A., Röder, M., Napolitano, G.: 7th open challenge on question answering over linked data (QALD-7). In: Dragoni, M., Solanki, M., Blomqvist, E. (eds.) SemWebEval 2017. CCIS, vol. 769, pp. 59–69. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69146-6_6
12. Berant, J., Chou, A., Frostig, R., Liang, P.: Semantic parsing on freebase from question-answer pairs. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1533–1544 (2013)
13. Bordes, A., Usunier, N., Chopra, S., Weston, J.: Large-scale simple question answering with memory networks. arXiv preprint arXiv:1506.02075 (2015)
14. Unger, C., Freitas, A., Cimiano, P.: An introduction to question answering over linked data. In: Koubarakis, M., et al. (eds.) Reasoning Web 2014. LNCS, vol. 8714, pp. 100–140. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10587-1_2
15. Höffner, K., Walter, S., Marx, E., Usbeck, R., Lehmann, J., Ngonga Ngomo, A.C.: Survey on challenges of question answering in the semantic web. Semantic Web **8**(6), 895–920 (2017)
16. Diefenbach, D., Lopez, V., Singh, K., Maret, P.: Core techniques of question answering systems over knowledge bases: a survey. Knowl. Inf. Syst. 1–41 (2017)
17. Xu, K., Feng, Y., Zhao, D.: Xser@ qald-4: answering natural language questions via phrasal semantic parsing. In: Working Notes for CLEF 2014 Conference, pp. 15–18 (2014)
18. Mazzeo, G.M., Zaniolo, C.: Answering controlled natural language questions on RDF knowledge bases. In: EDBT, pp. 608–611 (2016)
19. Zou, L., Huang, R., Wang, H., Yu, J.X., He, W., Zhao, D.: Natural language question answering over RDF: a graph data driven approach. In: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, pp. 313–324. ACM (2014)
20. Diefenbach, D., Singh, K., Maret, P.: WDAqua-core0: a question answering component for the research community. In: Dragoni, M., Solanki, M., Blomqvist, E. (eds.) SemWebEval 2017. CCIS, vol. 769, pp. 84–89. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69146-6_8
21. Lukovnikov, D., Fischer, A., Lehmann, J., Auer, S.: Neural network-based question answering over knowledge graphs on word and character level. In: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, pp. 1211–1220 (2017)
22. Jain, S.: Question answering over knowledge base using factual memory networks. In: Proceedings of the NAACL Student Research Workshop, pp. 109–115 (2016)
23. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The stanford CoreNLP natural language processing toolkit. In: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55–60 (2014)

24. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
25. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250. ACM (2008)