DBpedia and YAGO as Knowledge Base for Natural Language Based Question Answering—The Evaluation

Tomasz Boiński(✉) and Adrian Ambrożewicz

Department of Computer Architecture, Faculty of Electronics, Telecommunication and Informatics, Gdańsk University of Technology, Gdańsk, Poland
tobo@eti.pg.gda.pl

Abstract. The idea of automatic question answering system has a very long history. Despite constant improvement of the systems asking questions in the natural language requires very complex solutions. In this paper the DBpedia and YAGO are evaluated as a knowledge bases for simple class 1 and 2 question answering system. For this purpose a question answering system was designed and implemented. The proposed solution and the knowledge bases were evaluated and some remarks are given.

1 Introduction

The idea of automatic question answering system has a very long history. The first steps were performed back in late 60's of the previous century [12]. Over last decades human-computer interaction capabilities were constantly being improved [5,15]. Information retrieval and knowledge management is performed by increasingly complex systems. Thanks to rapid expansion of processing power, network and storage performance computers take part in information storage and processing on whole new levels. The idea of information retrieval however remains unchanged. We are required to provide some kind of a keyword set that will describe our query. The situation changes drastically with introduction of a queries formed using natural language. In this case there are no keywords to match and the query might not be explicit. During TREC conferences five levels of the questions complexity were defined and Moldovan et al. [11] formulated the requirements for each of those classes. Our interest lies in the first two classes: factoid questions (class 1), where the ability to answer facts usually requires finding direct answer in knowledge base in form of text snippet or database entry and reasoning and preprocessing based (class 2), where semantic knowledge is necessary both about question (structure of question provides information about point of interest and relations) and processing data sources (answer may not be syntactically close to a question). Other three levels (answering based on multiple sources (class 3), interaction-based (class 4) and expert systems able of analogical reasoning (class 5)) are considered for the future work.

DOI 10.1007/978-3-319-67792-7_25

The aim of this paper is to evaluate DBpedia and YAGO as a viable knowledge base for answering class 1 and class 2 questions. We wonted to evaluate those databases to check whether they are mature enough to create a valid solution that will be able to answer simple class 1 questions.

The structure of the paper is as follows. Section 2 describes different solutions for question answering and tools used in our approach. Section 3 presents our solution in detail. In Sect. 4 evaluation of the proposition is given. Finally in Sect. 5 some conclusions are given.

2 State of the Art

Question answering competition at TREC allowed multiple interesting solutions to emerge. The best performing TREC systems could answer approximately 70% of the questions from the TREC data set. Knowledge-rich approaches that used a vast array of NLP techniques were very successful in year 2000 [8]. Some, like AskMSR [2], proved that simple methods can be very successful. Recently two big systems emerged: Watson for English language and NEKST for Polish language. Both of those systems defined new standards in the field of natural language based question answering systems.

IBM Watson [7] is the worlds most famous supercomputer and set of algorithms aimed to answer natural language questions due to winning the Jeopardy TV show [6]. It is developed as part of DeepQA Project, which started in 2007. First Watson implementation was made using a cluster consisting of around 2500 CPUs, 15 TB of RAM and, with no connection to the Internet.

System itself is split into set of designated algorithms cooperating together to provide the best possible answer for given question and deep learning techniques are used to prepare the data accurate data retrieval possibilities.

The NEKST project realized by Polish Academy of Sciences with cooperation from Wrocław University of Technology [3] aimed at creation of intelligent search engine for polish Web resources. Question answering [9] is performed based on facts gathered by indexing of polish text resources available online and facts exploration. Algorithms developed for purposes of this project are oriented around contextual analysis of text resources in terms of facts, relations, sentimental bias, hierarchical ontologies and their massive parallelisation.

The above systems are very complex solutions that are only partially available for the general audience. Other systems are thus regarded as candidates for knowledge bases, like Wikipedia. It is however very informal and not standardized. Some teams tried to overcome this problem [14] and other, formalized versions exists in form of YAGO [13] and DBpedia [1]. In both cases data is extracted in automated way and represents in form of semantic triples. These triples can be then subject of SPARQL queries.

The DBpedia ontology consists of 492 classes with 53827 predicates. Empirical evaluation of random entries in database have shown that entities with the most unified feature sets are simple, well-defined concepts like: city, music album, language, country etc. Among entries of the same type they shown high level

of similarity. In other cases domain-specific facts are used. In case of human beings only some basic information like name, date of birth and nationality was available with all concepts. Many of the properties were derived only from info boxes, thus showing low level of linking with defined ontology.

Although DBpedia tries to fit into world of Linked Data it is poorly interlinked internally. In most cases the only aggregate for conceptually close entries are Wikipedia categories. DBpedia categories are converted into proper ontology entries but this is also an ongoing task.

YAGO is knowledge base developed under supervision of Max Planck Institue in SaarbrÃcken. Contrary to open-domain approach used in DBpedia, this is developed by limited number of dedicated people. One of project goals is to aim for accuracy rather than size of gathered resources. It is far more formalized than DBpedia and utilizes grouping categories ("Presidents of United States", "Born in 1990") as source of facts about underlying entities. Both category names and article titles also placed within WordNet taxonomy.

Accuracy of facts stored in YAGO data sets is constantly supervised with surveys performed on random sets of facts. Reports shows that mean level of accuracy is reaching 95% [13]. One of main assumptions in creation of YAGO ontology is distinction between absolute facts, and facts connected with time and place.

3 Towards Question Answering

3.1 The Goals

The primary goal of current work was to validate both YAGO and DBpedia data sets as knowledge bases for question answering systems. For this purpose we created class 1 and 2 question answering system, which task was to answer factoid questions about entities represented in the knowledge base. The idea is to give possibility of retrieving all known facts about chosen area of interest. User should be able to ask questions like "Who is author of 'Game of Thrones'?". More complex questions like "What is nationality of author of 'Game of Thrones'?" are also considered as being in reach of the currently proposed approach.

The assumptions that considered during creation of our approach were:

- Knowledge base independent—the system should be applicable for any existing knowledge base. Algorithms should not rely on particular "flavor" or data organization in database of choice.
- Change user queries to relations—a natural consequence of underlying data source. Question should be transformed into form of relations and than resolved with semantic database resources. As the facts are represented in knowledge base as triples, questions should be also transformed to form of triples with one unknown element to be resolvable with SPARQL query.
- Utilize knowledge base data on every level of processing – as we have no access to data sources other than the knowledge bases user query should be represented in terms of entities and relations known to semantic database. This

allows early optimization of queries and focusing on retrieving information that is actually possible to extract.
- Use semantic data features—usage of two interlinked, but still separate databases would not be possible if not for ontologies allowing correlation between those data sources.
- Focus on entity features and connections—the system should be able to solve problems in terms of retrieving certain property of entity under question or another entity correlated with it in well-defined relation. As a result factoid questions in form "What is [property] of [entity]" or "Who is [relation] of [entity]" are the main focus of our approach at this time.

3.2 Three Stages of Question Answering

Question answering consists of three stages: receiving input and query analysis, searching for user intentions, retrieving information.

Query Analysis—consists of retrieving user query and its initial analysis.

Grammatical Parsing—Stanford NLP Parser [4] is used for analysis of the structure of the question and provides output in form of ordered tree [10].

Lets consider the following query: *What is height of Eiffel Tower?* It is assumed that user input will be formulated in a way close to the example, allowing accurate clarification of user intentions. In this particular case - user intention is to retrieve height property of Eiffel Tower entity. This corresponds to querying underlying knowledge base with this specific SPARQL query:

$$select\ ?height\ where\ \{<EiffelTower><hasHeight>\ ?height\}.$$

When the query is converted into a tree structure (Fig. 1) the expected relation in the question is denoted as a set of parallel branches of the same subtree with distance from each other equal to one. This is further utilized in dependency parsing stage. Our experiments show that models provided with the parser are

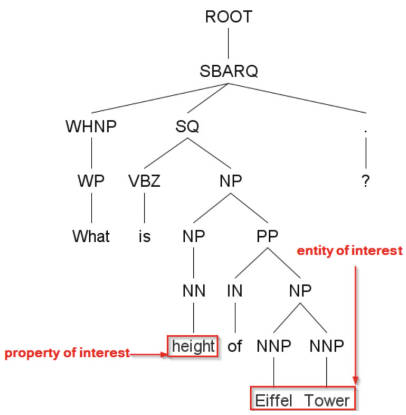

Fig. 1. Tree structure produced by Stanford NLP Parser, using Penn Treebank syntax

accurate in determining question structure for simple queries. Providing more complicated queries effects in incorrect parse results which are a primary way for further improvement of our solution.

Concept Retrieval—in this step we extract the concepts from parsed query sentence. This step is crucial as in the following stages only the extracted concepts will be subject of information retrieval. In our example the goal is to recognize both height property and Eiffel Tower entity.

Query Analysis Output—both structural parsing and concept retrieval provide basic information about the user's query. The output consists of:

- alternative versions of user query which express the same intention,
- Penn-Treebank tagged sentences in form of sentence structure trees,
- list of recognized concepts/named entities with a value of confidence level.

Query Break-Down—the task of this stage is to determine the entity of interest and its particular feature requested by the user. The relations within query are analyzed to approximate which part of query is the focus of question.

Dependency Parsing—with the query grammatically parsed it is broke down into basic semantically connected concepts. The dependency parser logically binds the concepts and determines relation between them creating a dependency graph. The graph is conceptually close to grammatical sentence structure but omitting syntactical alterations not relevant to the meaning of the query. For the query "Who is author of 'Game of Thrones'?" the raw output from dependency parser is as follows: cop (Who, is), nsubj (Who, author), nmod (Game, author), case (Game, of), nmod (Game, Thrones), case (Thrones, of).

Two tasks are required during this processing stage: correlating dependency parser output with results of grammatical and conceptual parsing done in previous stage and building dependency graph of the user's query.

Minimization of Dependency Parser Output—only meaningful relations found by the dependency parser are taken into further consideration. The set produced by our algorithm is than converted into a tree. We assume that the question will have only one root determined during relation processing. Starting from the root, relations are joined together to form dependency graph, as shown in Fig. 2.

Analyzing Dependency Tree: Generating 'Fibers'—during our work we observed that dependency trees of most of the queries had one or more paths leading from the root to named entity of interest. These paths (referenced further as fibers) are candidates for representation of actual user intention. The goal of this step is to retrieve minimal fiber (as marked in Fig. 2 in red).

This process usually minimizes graph to form of few fibers which will be then subject for final stage of information retrieval. In this particular case only path "Who > author > Game of Thrones" is generated.

Query Break-Down Output—dependency parsing combined with data retrieved in previous step effectively produces final input for information retrieval subsystem. This data consists of fibers which are minimized to contain only meaningful

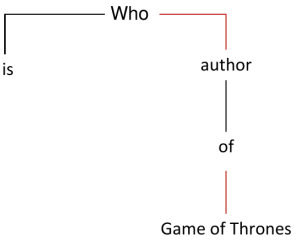

Fig. 2. Final result of building dependency graph. Path of interest is colored in red.

data required for retrieving triples from knowledge base. The fibers are ranked based on their length and number of known concepts, the one with shortest path leading to concepts in question is the most probable candidate for actual user intention representation.

Information Retrieval—fibers from previous step are sorted based on their score of relevancy and evaluated in order from most relevant to least. The first answer retrieved from the database is presented to the user. Example graphical representation of answer retrieval is shown in Fig. 3.

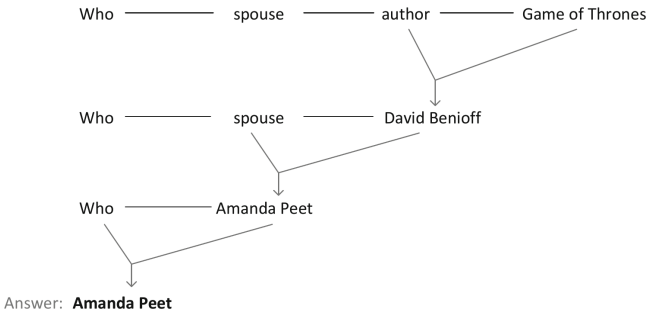

Fig. 3. Answer retrieval for question "Who is spouse of 'Game of Thrones' author?"

4 Evaluation

4.1 Test Questions Set

The solution was created to evaluate usability of YAGO and DBpedia for answering factoid questions. Aforementioned classification proposed by Moldovan places it in Class 1. Limited possibility of answering certain Class 2 questions was also considered. To evaluate the solution a subset of 200 original questions used during TREC-8[1] was selected. The set consists of 40 well-formed questions with intention to retrieve basic properties and relations.

[1] http://trec.nist.gov/data/qa/T8_QAdata/topics.qa_questions.txt.

4.2 Test Results

Each question was subject of information retrieval process which was evaluated in the following terms: availability of information in database, correct answer retrieved by the system (true positive), incorrect answer retrieved by the system (false positive), interpretation of failure result with optional cause details. Summary in terms of answer correctness and performance evaluation using precision, recall, true negative rate, accuracy and F-measure is presented in Table 1.

Table 1. Answer correctness and performance scores

True positives	True negatives	False positives	False negatives	Recall	Precision	Specificity	Accuracy	F-Measure
8	18	4	14	0.36	0.67	0.82	0.59	0.47

Achieved results looks promising, but only with consideration of fact, that test set was already tailored down to set of simple factoid questions. Various kind of issues were observed in every subsystem. In evaluated test set following classes of problems occurred:

- relation predicate similarity—both place and date of birth were returned when only one of them was required,
- relation not mapped correctly—answer was available but under some syntactically distant name (e.g. "leadername" instead of "president"),
- entity recognition error—entity under question was available in database but due to parsing issues it was not correctly detected,
- property not available—entity was correctly recognized, but knowledge bases didn't contain expected information,
- entity under question not available—knowledge base didn't contain any explicit information about entity under question.

4.3 Result Analysis

Based on test results certain gaps were identified in our solution. They are presented in the following sections.

Stage One: Query Analysis—the algorithm is highly relaying on accuracy of dependency parser. When the questions are simple and well-formed, it worked well. However, there are several issues being consequences of the quality of the parser used.

Question Reformulation—some forms of the questions were difficult to transform to the correct query (i.e. "what is height of" vs. "how tall is") or the questions contained synonyms not detected by the parser (height vs. elevation).

Addressing majority of this issues requires high-tier linguistic processing. Utilizing available set of tools, the following problems can be addressed:

– for word alternative forms (height vs high) WordNet may be used to generate alternative queries for the parser,
– changed order of words in compound clauses (named entities) may be addressed with stemming and permuting sets of words,
– stemming of words alongside with wild card/regular expression search in knowledge base may be solution for varying form of predicates,
– syntactic databases may be used to determine equivalent relations.

Parser Quality—it is crucial in our approach. In several cases parser failed to distinguish named entities from other parts of sentence.

Ambiguity of Concepts—some concepts are represented in knowledge base as multiple separate entities. For example concept "Abraham Lincoln" has 6 separate meanings, including president, bridge and a car. For best results, even partial matches of entity names are considered. When certain threshold of certainty is reached, retrieved entity is treated as one of the candidates in information retrieval step. Confidence level is measured with string similarity methods, as WordNet similarity metrics does not apply for named entities.

Stage Two: Query Break-Down
Parser Accuracy—the parser performs well in regard to class 1 questions, for more complex forms of question a more robust solution is needed.

Dependency Direction—in most cases dependency between words was correctly determined without the consideration about the direction. This is however not a problem as the fibers are not direction dependent.

Stage Three: Information Retrieval
Focus on Question Root—for higher tier of question answering the system should be able to represent the question by multiple fibers.

4.4 Areas of Further Development

Our solution was developed for evaluation of the YAGO and DBpedia and as such is considered as entry-level solution for question answering systems. It is able to answer factoid questions based on semantical knowledge base. Major areas of improvement, required to create system of higher class, in correspondence to described three steps model, are the following:

User Input Analysis—In the next step the system should be extended to answer higher tier of questions like "What is difference between barter and trade?" (relations), "What diseases are specialized at attacking lungs?" (lists) or "President of which country was Abraham Lincoln?" (reverse-order questions). Implementing support for these classes of problems would be sufficient to prepare true Class 2 solution. We believe that the chosen knowledge bases should be able to answer such questions.

Another limitation was the parser used. When it comes to general-purpose utilization it may be useful, but for question answering more specialized solution is required. Learning parser with actual user questions corpora could provide major improvement in area of correct determination of query structure.

Named entity recognition could also be improved. In our approach they are recognized using text search in underlying knowledge bases. Incorporating dedicated tools as helpers for that matter may bring large improvements to the system. Also spelling errors should be handled before the query is analyzed.

Answer Retrieval—the evaluated knowledge bases are very big in size and contain information about large number of entities. However, this knowledge is often limited to short list of properties for particular entry. Furthermore the number and character of these properties differ not only between different kind of concepts (e.g. people and events) but between supposedly similar ones.

5 Conclusions

Prepared system have shown that it is possible to create working open-domain question answering system with freely available tools and data sets. Proposed approach is working well for retrieval not only simple facts of certain entities, but have possibility to solve compound, internally dependent queries, as long as they follow relation-based querying model.

With certain improvements, such as support for different kinds of questions and preparing fine-tuned support for ontologies in underlying knowledge bases it has chances to become Class 3 system, able to answer questions requiring utilizing certain extent of reasoning. The proposed query analysis techniques may also serve the purpose of semantic data extraction from text. Concept of mapping user query to relations in underlying knowledge base with some adjustments may be used to create relations based on parsing news articles or Wikipedia pages.

However, it is expected that with current state of available knowledge bases it's not possible to go beyond that level. The most notable reason is lack of representation of complex issues. Currently available open-domain RDF resources are usually limited to simple facts and relations. Ontologies themselves may serve purpose of information retrieval techniques but task of utilizing them is very hard as it requires specialized, possibly deep-learning, algorithms aimed to solve particular fine-grained problems as consecutive steps for building higher level solution. Distinction between ontologies also provides certain level of complication as solution applicable for one ontology may be useless for the other.

Idea of Linked Data and Semantic Web couldn't be utilized directly with evaluated databases. Their ontology is very shallow, specialized and not well interlinked. Data linking occurs usually at subject level, but even that is not always the case. To be able to prepare robust open domain question answering system underlying data should be linked in a way, which should allow for universal methods of data exploration, independent on data contributors.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K. (ed.) The Semantic Web. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
2. Banko, M., Brill, E., Dumais, S., Lin, J.: AskMSR: question answering using the worldwide web. In: 2002 AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases, Palo Alto, USA, pp. 7–9 (2002)
3. Czerski, D., Ciesielski, K., Dramiński, M., Kłopotek, M., Łoziński, P., Wierzchoń, S.: What NEKST?—Semantic search engine for Polish internet. In: Tre, G., Grzegorzewski, P., Kacprzyk, J., Owsiński, J., Penczek, W., Zadrożny, S. (eds.) Challenging Problems and Solutions in Intelligent Systems. SCI, vol. 643, pp. 335–347. Springer, Cham (2016)
4. De Marneffe, M.C., MacCartney, B., Manning, C.D.: Generating typed dependency parses from phrase structure parses. In: LREC 2006, Geona, Italy, vol. 6, pp. 449–454 (2006)
5. Duch, W., Szymański, J., Sarnatowicz, T.: Concept description vectors and the 20 question game. In: Kłopotek, M., Wierzchoń, S., Trojanowski, K. (eds.) Intelligent Information Processing and Web Mining. AINSC, vol. 31, pp. 41–50. Springer, Heidelberg (2005)
6. Ferrucci, D., Levas, A., Bagchi, S., Gondek, D., Mueller, E.T.: Watson: beyond jeopardy!. Artif. Intell. **199**, 93–105 (2013)
7. Ferrucci, D.A.: IBM's Watson/DeepQA. ACM SIGARCH Comput. Archit. News **39**(3) (2011)
8. Harabagiu, S.M., Moldovan, D.I., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R.C., Girju, R., Rus, V., Morarescu, P.: FALCON: boosting knowledge for answer engines. In: TREC 2000, Gaithersburg, USA, pp. 479–488 (2000)
9. Marciriczuk, M., Ptak, M., Radziszewski, A., Piasecki, M.: Open dataset for development of Polish question answering systems. In: LTC 2013, Poznań, Poland (2013)
10. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: the penn treebank. Comput. Linguist. **19**(2), 313–330 (1993)
11. Moldovan, D.I., Harabagiu, S.M., Paşca, M., Mihalcea, R., Goodrum, R.A., Gîrju, C.R., Rus, V.: LASSO: a tool for surfing the answer net. TREC **1999**, 65–73 (1999)
12. Simmons, R.F.: Natural language question-answering systems: 1969. Commun. ACM **13**(1), 15–30 (1970)
13. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: WWW 2007, pp. 697–706. ACM, Banff (2007)
14. Szymański, J.: Self-organizing map representation for clustering wikipedia search results. In: Nguyen, N.T., Kim, C.G., Janiak, A. (eds.) Intelligent Information and Database Systems. LNCS, vol. 6591, pp. 140–149. Springer, Heidelberg (2011)
15. Szymański, J.: Words context analysis for improvement of information retrieval. In: Nguyen, N.T., Hoang, K., Jedrzejowicz, P. (eds.) International Conference on Computational Collective Intelligence. LNCS, vol. 7645, pp. 318–325. Springer, Heidelberg (2012)