SemGraphQA@QALD-5: LIMSI participation at QALD-5@CLEF

Romain Beaumont (1,2), Brigitte Grau (1,3) and Anne-Laure Ligozat (1,3)

(1) LIMSI-CNRS, (2) Universite Paris-Sud, (3) ENSIIE, France
{firstname.lastname}@limsi.fr

Abstract. For our participation to QALD-5, we developed a system for answering questions on a knowledge base. We proposed an unsupervised method for the semantic analysis of questions, that generates queries, based on graph transformations, in two steps. First step is independent of the knowledge base schema and makes use of very general constraints on the query structure that allows us to maintain semantic ambiguities in different graphs. Ambiguities are then solved globally at the final step when querying the knowledge base.

Keywords: Semantic Web, Question-Answering system, Semantic question analysis.

1 Introduction

More and more structured knowledge is made available on the Web, as with DBpedia [Auer et al., 2007], Freebase [Bollacker et al., 2008] or Yago2 [Hoffart et al., 2011], allowing a user to find answers related to some precise information need. To enable a user to have simple access to these bases, simple interfaces that support questions given in Natural Language (NL) are developed, which do not require a user to know how the knowledge is structured, i.e. the knowledge base schema, and to learn a formal language, e.g. SPARQL. A SPARQL query relies on a conjunction of triples that can be represented as a graph: nodes are entities and edges are typed relations. The two interrelated problems we are faced with when transforming a NL question into such a graph are: i) identifying entities, relations, entity classes and operators and ii) determining the structure of the query, i.e. how these entities and relations are related and how the operators apply. [Unger et al., 2012] and [Xu et al., 2014] make first structural choices based on the knowledge base schema before knowledge base mapping, while [Yahya et al., 2013] and [He et al., 2014] solve all the ambiguities together.

We chose to develop an unsupervised method based on graph transformation that acts in two steps: first, syntactic graphs are transformed for building all the structural valid syntactico-semantic representations of the question that maintain ambiguities related to the queried knowledge base schema and content. The second step consists in generating all the semantic graphs made of valid triples, determined according to the knowledge base schema and content, i.e. DBpedia, each leading to a SPARQL query. These ambiguities are solved at the last step, close to [Zou et al., 2014]. Each semantic graph is weighted according to scores attributed to triples, and the answer is the result of the top ranked query that applies.

The system, named SemGraphQA, was designed for answering DBpedia queries, and cannot answer hybrid ones. It obtained a F1 of 0.31 at QALD-5.

2 Description of the system

The aim of the system is to transform a question Q into semantic graphs G_Q^S, each corresponding to a possible meaning. The nodes of the graph are entities and the edges are relations. Every entity and relation has a relevance score, which is used to compute the score of the graph. This transformation involves different steps (Figure 1).

Question phrases are identified in order to link them to semantic items of the knowledge base: entities, types or relations (Figure 1a). Entity identification is performed by DBpedia Spotlight [Daiber et al., 2013], type identification is done using the type labels, and relation identification by a method we propose based on variations extracted from WordNet. To each possible semantic item a relevance score is given.

The syntactic analysis of the question by the Stanford Parser [Klein and Manning, 2003] produces a syntactic graph which is simplified and rewritten to generate a G_Q^{Sy} graph (Syntactic Graph of the Question) which has for nodes the words and for edges the syntactic relations. In that graph, the semantic nature of each word (entity or relation) is unknown (Figure 1b).

The G_Q^{Sy} graph is multiplied by assuming each word can be either entity or relation in order to keep the possible ambiguities, and then transformed and filtered in order to obtain the graphs G_Q^{SS} which comply to structural constraints (Figure 1c). For example, if two *relation-word* are linked by a syntactic relation, it means there is an implicit entity and we add an *entity-word* between them. In the question *Who is the daughter of Bill Clinton married to?*, when *daughter* and *married* are considered as *relation-word*, they are linked directly, so we create an *entity-word* unknown between the two, which explicits the fact to have to find the daughter of Bill Clinton to answer to the question.

The last step consists in generating the possible semantic graphs, noted G_Q^S, using the ambiguities of each node and edge obtained after the mapping phase. These graphs are constituted of triples consistent with the knowledge base (Figure 1d). These semantic graphs are sorted based on a relevance score and the corresponding queries are executed in descending order until at least one answer is returned.

The method we propose aims at never eliminating an interpretation that has a coherent semantic structure, i.e. a structure that can be instantiated in the queried data. The constraints on the graph structure are very general and not specific to the schema of a given knowledge base. Thus, these constraints could also be used when searching information in texts. The constraints linked to the

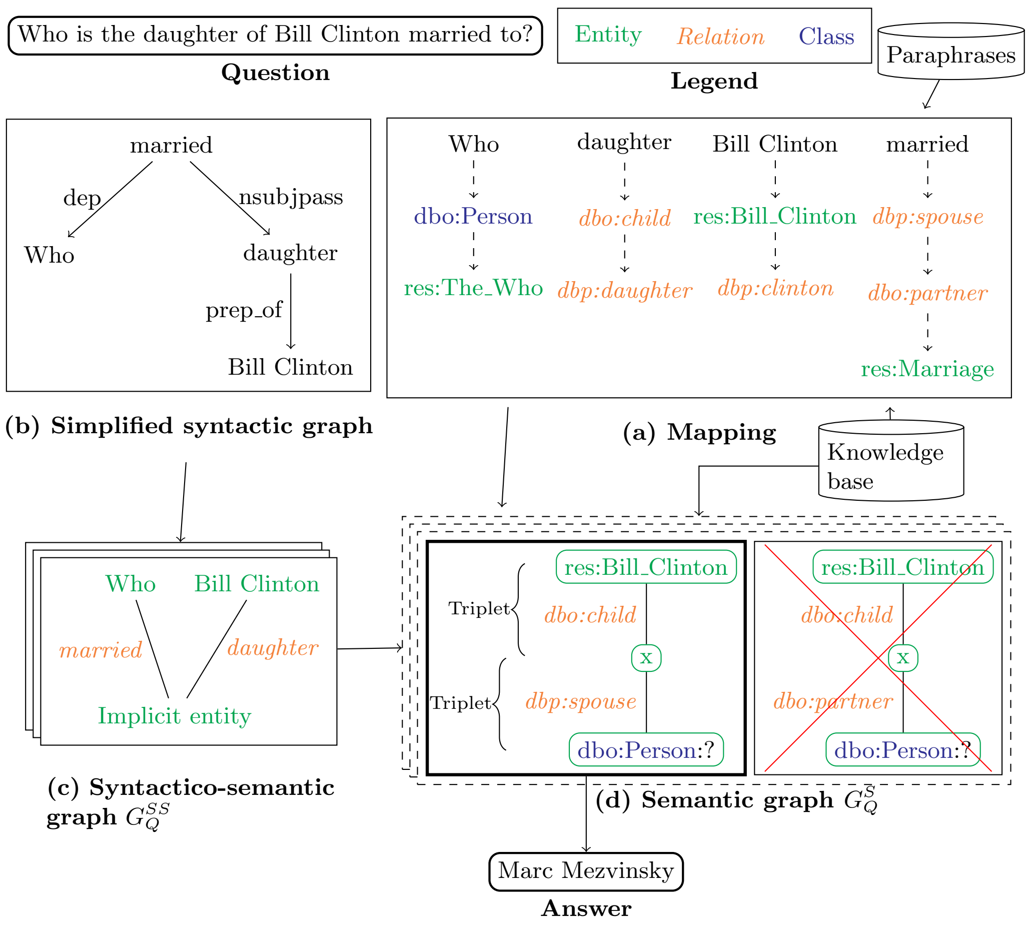

Fig. 1. System overview

queried base are applied at the last step and the final filtering is operated by the search in the knowledge base, based on finding an answer or not.

3 Semantic component mapping

In order to map question terms to semantic components of the knowledge base (KB), i.e. entity, relation and class (cf. Fig. 1a), we determine a set of phrases that could be related to a semantic component according to its label. These phrases correspond to all n-grams, with n maximum 5, that contain a noun, a verb or an adjective, as each type of word may lead to a type of component. A reliability score is associated to each found component.

3.1 Entity extraction and identification

Entity mapping consists of recognizing an entity mention in a question, i.e. a word or a phrase that refers to an entity, linking it to a KB resource and determining its class. To this purpose, we make use of DBpedia Spotlight (DBS) [Daiber et al., 2013]. DBS determines possible entities, extracts their mention, associates them with a score, and classifies them according to the DBpedia ontology[1] that contains a hierarchy of classes. We use the DBS score as the reliability score of an entity. It is based on a score computed during the indexing of the entities and contextual scores depending on the other question terms. As we do not want to choose between entities at a early stage, we keep all results provided by DBpedia Spotlight.

3.2 Class identification

Some question phrases refer to classes, and not to entities. For example, in the question *Which languages are spoken in Estonia?*, *languages* corresponds to the class *dbo:Language* and in *Which software has been developed by organizations founded in California?* , *organizations* corresponds to the class *dbo:Company* and *software* to the class *dbo:Software*.

The identification of classes enables to further associate it with the domain or range of a relation. Target classes are those of DBpedia classes and Yago classes[2]. Their identification is based on a comparison of candidate n-grams to class labels. Recognized classes are considered as unknown entities that could be found when querying the KB.

Scores of recognized classes are based on the Jaccard distance of matching words to favour specific classes. For example, the class *StatesOfTheUnitedStates* will obtain a greater score than *States*. Thus, the score *Sclass* for evaluating the reliability of a class is:

$$Sclass = \alpha * \frac{\#\ words}{maximum\ number\ of\ words\ in\ a\ label} \tag{1}$$

[1] http://mappings.dbpedia.org/server/ontology/classes/

[2] DBpedia includes classes of Yago [Hoffart et al., 2011]

We set maximum to 10. As DBpedia classes are less numerous than Yago classes (529 vs 379900), and more often present in the DBpedia triples, we set a parameter α to 2 for DBpedia classes and 1 for Yago to lower Yago classes score.

A special case is due to interrogative words. They are processed separately and each interrogative word is mapped to a class, for example *Who* is mapped to *dbo:Agent* and *Where* is mapped to *dbo:Place*.

3.3 Relation identification

Identification of a relation consists of associating a mention extracted from the question to a KB relation. Each KB relation is associated with a label, which corresponds to a possible mention in texts. In order to maximize recall for relations, we depart from all candidate mentions, made of the different question n-grams, rather than pre-filtering them according to a classification into relation and non relation classes.

However, few relations are mentioned in questions using their KB label and several kinds of variation have to be considered:

1. The POS category of the word in the label is different from the POS category of the word in the mention. For example, in *Who is Barack Obama married to ?*, the verb *married* has to be related to the relation with a label made of the noun *spouse*.
2. Mentions of relation and labels are paraphrases or present a greater semantic distance. For example a mention can refer to an hyponym as in Figure 1a.
3. Labels and mentions are made of several words, i.e. n-gram with n greater than 1, and each of the words can present variations.

In order to overcome this lexical gap, we built a lexicon with variants acquired from WordNet [Fellbaum, 1998]. We selected the following relations in WordNet:

- derivationally related forms: *successor* $\leftrightarrow$ *succeed*
- synonyms: *author* $\leftrightarrow$ *writer*
- hypernyms/hyponyms : *relative* $\leftrightarrow$ *sister*

The algorithm for acquiring variants depart from a label. For each word, the network is recursively explored at a maximum depth d. We limit the depth in order to compute the most relevant variants. We experimentally fixed $d = 4$.

Each variant is assigned a score based on the path length:

$$S_r = \frac{1}{\textit{path length between two terms}} \tag{2}$$

with the path length equal to the number of crossed hypernyms/hyponyms links.

We built a dictionary of variations for all the relations in DBpedia. These relations all have an English label, for example *dbo:child*, which indicates all the children of a person, has the label *child* and *dbo:birthDate*, which indicates the birthday of a person, has the label *birth date*. There is a total of 1,252,327 variations generated for 12,331 relations, with a mean of 102 variations by relations.

4 Generation of a semantic representation of a question

4.1 Question Semantic Representation

Each possible meaning of a question is represented by a set of triples (e_1, R, e_2), with e_1 and e_2 entities and R a relation, making a connected and weighted semantic graph G_Q^S. Entities are linked by a relation only if a syntactic relation exists between their mentions. Triples are those which have an instantiation in the KB.

An entity is described by four fields: *mention*, *type*, *value* and *score*. The fields *mention* and *type* can be empty. For example an interrogative word is an entity with only a *type* and a *label*. The *value* field is an *uri*, a variable x when it is an implicit entity in the question, or ? when it is the entity to find. The entity score is computed by the mapping process (cf Section 3.1), and has a value when an *uri* is found.

A relation has a label (in the knowledge base), a domain, a range and an identifier in the knowledge base (uri). For example the relation *dbo:birthDate* (*http://dbpedia.org/ontology/birthDate*) has the label *birth date*, the domain *dbo:Person* and the range *xsd:date*

In case of ambiguous question terms, that have been related to different elements of the knowledge base, several semantic graphs will be generated for representing the different meanings of the question.

4.2 Triple Generation

Triples are generated from the syntactic dependency graph given by the Stanford parser [Klein and Manning, 2003]. This process is decomposed into two stages: the production of syntactically well-formed syntactico-semantic graphs G_Q^{SS}, and the generation of semantically coherent semantic graphs G_Q^S.

The transformation of a syntactic graph into a G_Q^{SS} graph is done by the following steps:

1. the syntactic graph is simplified to keep only the relevant dependency relations. The syntactic relations such as determiner, noun compound modifier, controlling subject, passive auxiliary, auxiliary, open clausal complement, controlling subject and copula are filtered. The remaining relations form a graph of words (see Figure 1b);
2. a word may refer to an entity or a relation. This entails several possible graph structures and several graphs are generated for every possibility with each term replaced by an $entity - word$ or a $relation - word$ node;
3. these graphs are then filtered by well-formedness criteria: each $entity - word$ must be linked to a $relation - word$; if two $relation - word$ are linked, an $entity - word$ corresponding to an implicit entity is added; every $relation - word$ must be linked to two $entity - word$s; the graph must be connected (cf Figure 1c).

Once these G_Q^{SS} graphs of words typed as relation or entity are generated, it is possible to generate the semantic graphs G_Q^S constituted of semantic triples:

1. every triple element is mapped to each semantic item found during the mapping step. It makes it possible to generate different semantic triples for each triple of a G_Q^{SS} graph;
2. the semantic coherence of each triple is checked by keeping only the triples which have an instantiation in the knowledge base;
3. the Cartesian product between the remaining triple lists is computed to obtain the semantic graphs G_Q^S, which are the possible semantic representations of the question (cf Figure 1d).

Generating G_Q^{SS} graphs before generating G_Q^S graphs enables to filter a subset of the graphs with structural criteria, without taking into account the mapping ambiguities on each node and edge, and hence produce less semantic graphs.

4.3 Incompatible triple identification

Once all the possible triples are known, it is possible to evaluate their consistency relative to semantic constraints provided by the KB schema. In KB triples, each of the two entities can be associated to a type, and relations are defined by a domain and a range. Using this information, we can compute a compatibility score by giving a lower score to triples that are not compatible, i.e. the types of the joint entities do not fit domain and range. Indeed we cannot simply dismiss the incompatible triples because the knowledge base may be inconsistent: some entities are linked by a relation without observing the domain or range constraints. For example, the answer to the question *Who developed Minecraft?* corresponds in DBpedia to the triple ⟨*res:Minecraft dbo:developer res:4J_Studios*⟩. The domain of *dbo:developer* is *dbo:UnitOfWork* and the range is *dbo:Agent*. *res:Minecraft* has type *dbo:VideoGame* which is not compatible with *dbo:UnitOfWork*.

We define the compatibility between an entity e of type C_e and a domain D of type C_D as: e is incompatible with D only if C_D is not included in the set formed of C_e and of its supertypes and subtypes. There are three possibilities: compatibility, incompatibility and the case where the information is missing if the domain or entity types are unknown. For example the relation *dbo:childOrganisation* identified as a candidate relation for the mention *daughter* between *Bill Clinton* and his daughter (implicit entity mention for which the type is unknown) is incompatible because it has as range *dbo:Organisation* which is not compatible with the type *dbo:President* of *Bill Clinton*.

This compatibility verification makes it possible to compute a score Sc of compatibility of a triple:

$$Sc = Scd \times Scr \tag{3}$$

with Scd and Scr the compatibility scores of the domain and range. They take the value β in case of compatibility, γ in case of missing information and δ in case of incompatibility. A compatible domain, or range, must obtain a better score than missing information or incompatibility, so we take $\beta > \gamma > \delta$ and we set $\beta = 1.0$, $\gamma = 0.75$ et $\delta = 0.5$.

4.4 Weighting of a semantic graph

Each semantic graph is assigned a weighting score S given by the formula:

$$S = \frac{\sum\limits_{i=1}^{n} St_i}{n} \tag{4}$$

with n the number of triples in the graph, and St_i the score of the triple i

$$St_i = Sc\frac{Se_1 + Sr + Se_2}{3} \tag{5}$$

with Se_1 and Se_2 the scores of the two entities, Sr the score of the relation and Sc the compatibility score of the triple.

The graphs are then converted to SPARQL queries and executed in descending order of their scores.

4.5 Question types and operators

The SPARQL query may involve the use of operators. Thus, we developed a basic method to handle some cases. The following question types are identified:

– count question : question starting by *How many* and that either needs a COUNT operator in the SPARQL query, or a relation usually containing word such as *count* or *total* in their label;
– boolean question: question usually starting by *Is* or *Are* and that are answered using a ASK type of query;
– factual question: all the other questions, that are answered by SELECT queries.

The question type is identified using a simple keyword matching. The same analysis is then done for all three kinds of questions, only the generation of the SPARQL query differs a little to handle the difference in the expected answer.

In several questions, binary operators are present. The operators can be for example *more than*, *less than*, *the same as* and are expressed in SPARQL queries as FILTER. We currently handle *more than* and *less than* operators by considering them as a special kind of relation. These relation are processed the same way as the knowledge base relation, until the transformation in SPARQL queries where they are represented as FILTER. For example the question *Which caves have more than 3 entrances?* is expressed by the query *select distinct ?2 where* {*FILTER(?1 > 3). ?2 dbp:entranceCount ?1. ?2 rdf:type dbo:Cave.*}

Some binary operators are not handled yet such as *same-as* operators, neither are unary operators represented in SPARQL as ORDER BY and LIMIT that correspond to superlative in the question such as "oldest child".

5 Experiments

The results in Table 1 are obtained on the QALD-5 test question set. It is made of 60 questions, with 50 questions on DBpedia and 10 hybrid questions. Our system is evaluated on the 50 questions on DBpedia.

	Processed	Right	Partial	Recall	Precision	F1
Xser	42	26	7	0.72	0.74	0.73
APEQ	26	8	5	0.48	0.40	0.44
QAnswer	37	9	4	0.35	0.46	0.40
SemGraphQA	31	7	3	0.32	0.31	**0.31**
YodaQA	33	8	2	0.25	0.28	0.26

Table 1. QALD-5 test evaluation

Among the 39 questions that the system does not answer correctly, there are overlapping errors in 26 questions: an entity is not found in 7 questions, a relation is not found in 25 questions and a type is not found in 4 questions. In the 13 remaining questions with errors, there are some issues such as the order operation (usually represented as a superlative in the question), and some binary operators such as *same-as* which are not handled by the system currently. An other error cause is an incorrect ranking of semantic graphs.

The main difficulty relies in the identification of relations, due to the lexical distance between the relation mention and its label in the knowledge base, or to a relation wordings in the question that does not make use of a content word.

6 Conclusion

In the context of systems enabling to answer questions on knowledge bases on the Semantic Web, we developed SemGraphQA, a system based on graph transformations designed for handling structural and lexical ambiguities. It enables to postpone choices until the latest point while limiting the number of ambiguities to consistent ones with graph structure. The method we used is unsupervised and can be easily applied to knowledge base other than DBpedia. Moreover, the semantic representation generated is independent of the query language.

We plan to build a hybrid system on knowledge base and texts in order to take into account the information present in textual corpora using the same semantic representation generation method with an adaptation of the identification of semantic items.

References

[Auer et al., 2007] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). *Dbpedia: A nucleus for a web of open data.* Springer.

[Bollacker et al., 2008] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). 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.

[Daiber et al., 2013] Daiber, J., Jakob, M., Hokamp, C., and Mendes, P. N. (2013). Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*.

[Fellbaum, 1998] Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Bradford Books.

[He et al., 2014] He, S., Liu, K., Zhang, Y., Xu, L., and Zhao, J. (2014). Question answering over linked data using first-order logic. In *Proceedings of Empirical Methods in Natural Language Processing*.

[Hoffart et al., 2011] Hoffart, J., Suchanek, F. M., Berberich, K., Lewis-Kelham, E., De Melo, G., and Weikum, G. (2011). Yago2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World wide web*, pages 229–232. ACM.

[Klein and Manning, 2003] Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics.

[Unger et al., 2012] Unger, C., Bühmann, L., Lehmann, J., Ngonga Ngomo, A.-C., Gerber, D., and Cimiano, P. (2012). Template-based question answering over rdf data. In *Proceedings of the 21st international conference on World Wide Web*, pages 639–648. ACM.

[Xu et al., 2014] Xu, K., Feng, Y., and Zhao, D. (2014). Xser@ qald-4: Answering natural language questions via phrasal semantic parsing.

[Yahya et al., 2013] Yahya, M., Berberich, K., Elbassuoni, S., and Weikum, G. (2013). Robust question answering over the web of linked data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1107–1116. ACM.

[Zou et al., 2014] Zou, L., Huang, R., Wang, H., Yu, J. X., He, W., and Zhao, D. (2014). Natural language question answering over rdf: A graph data driven approach. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 313–324, New York, NY, USA. ACM.