On the SPOT: Question Answering over Temporally Enhanced Structured Data

Mohamed Yahya Klaus Berberich Maya Ramanath† Gerhard Weikum

Max Planck Institute for Informatics, Germany †Dept. of CSE, IIT-Delhi, India
{myahya,kberberi,weikum} @mpi-inf.mpg.de ramanath@cse.iitd.ac.in

ABSTRACT

Natural-language question answering is a convenient way for humans to discover relevant information in structured Web data such as knowledge bases or Linked Open Data sources. This paper focuses on data with a temporal dimension, and discusses the problem of mapping natural-language questions into extended SPARQL queries over RDF-structured data. We specifically address the issue of disambiguating temporal phrases in the question into temporal entities like dates and named events, and temporal predicates. For the situation where the data has only partial coverage of the time dimension but is augmented with textual descriptions of entities and facts, we also discuss how to generate queries that combine structured search with keyword conditions.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval; I.2.1 [**Artificial Intelligence**]: Natural language interfaces

Keywords

question answering, structured data, knowledge bases, time

1. INTRODUCTION

Semantically rich structured data is ubiquitous. While such data is often used to support knowledge management tasks, there is a great value in having humans being able to directly access the wealth of knowledge in such data. Natural language questions provide a convenient way of doing this, as this relieves the end user from the need to be technically proficient in formal languages to pose a structured query, and from the need to understand the underlying schema of the data. Structured data often comes with a temporal dimension stating when events happened, or entities took on a particular state. Therefore, it is crucial that such question answering systems be able to deal with temporal questions and properly translate them to structured queries that yield the desired answers.

TAIA'13 August 1, 2013, Dublin, Ireland.

The association of time with structured data can be achieved in one of two ways: structured temporal annotations, or through textual resources. Structured temporal annotations provide us with well defined semantics. Textual documents with temporal expressions can also provide us with valuable temporal information. Such textual documents can, for example, give us a clue that the 1970's is an important time period for the movie Jaw, while the 1990's is an important time period for the movies Jurassic Park and Saving Private Ryan, allowing us to get an answer for the question "Which movies did Steven Spielberg direct after Jaws?"

As Table 1 shows, temporal questions show up in multiple question answering benchmarks. The questions vary in their invocation of time, from those that ask about times of specific events, to those that require an ordering of events.

This paper starts by outlining the data model that we base our work on, with facts associated with both a temporal and textual dimension. We then give a brief overview of an existing system capable of answering questions over structured data, to which we would like to add support for temporal questions that make use of temporal annotations.

We next show the pre-processing steps required to work with temporal questions and data in our setting. We then present a solution for the case where structured temporal annotations are available to fully answer a question, and the question-to-query translation works perfectly to allow for this. Finally, we consider the more realistic case where we have to rely on a mixture of both structured and unstructured (textual) time annotations to answer temporal questions, and the important role of ranking in this setting.

2. DATA MODEL

Our basic data model is that of RDF, with facts comprised of a subject (S), predicate (P), and an object (O). We refer to this model as the *SPO* model. Subjects and objects can be entities (`Jaws`), or classes (`Movie`). Predicates provide both instance (`directed`) and ontological information (`subClassOf`, `type`), with instance predicates having a type signature over classes. We call a set of such facts a knowledge base, examples of which include DBpedia [2], Yago [6], and Freebase [4].

This basic model can be enhanced with temporal information about facts, by adding an additional temporal dimension (T) that specifies a time interval, resulting in *SPOT* quads. The interval can be of length 0, indicating a time point, or it can be a proper interval. The special time point *now* can serve as the endpoint of an interval, indicating that a fact still holds. Finally, a fact can have a *null* interval, indicating that no time interval is known for the fact. This

1. When was Alberta admitted as province?	QALD
2. Who is the youngest player in the Premier League?	QALD
3. Which U.S. state has been admitted latest?	QALD
4. Was the Cuban Missile Crisis earlier than the Bay of Pigs Invasion?	QALD
5. Show me all songs from Bruce Springsteen released between 1980 and 1990.	QALD
6. Which movies did Sam Raimi direct after Army of Darkness?	QALD
7. For which label did Elvis record his first album?	QALD
8. What positions did Michael Brown hold before becoming head of FEMA?	TREC
9. What position did Obama hold before becoming US senator?	TREC
10. What Chinese Dynasty was during 1412-1431?	TREC
11. What President became Chief Justice after his presidency?	TREC
12. Who was Secretary of State during the Nixon administration?	TREC
13. Which Country has become independent after 30 years of civil war?	CLEF
14. In which region of Scotland was the proportion of crimes detected 31% during the period 1991-1993?	CLEF

Table 1: A sample of temporal questions

S	P	O	T
SyrianCivilWar	type	Event	[2011-03-15,now]
SyrianCivilWar	startedOn	2011-03-15	[2011-03-15,2011-03-15]
WorldWarI	type	Event	[1914-07-28,1918-11-11]
ChristianWulff	type	President	[2010-07-02,2012-02-17]
ChristianWulff	electedOn	2010-06-30	[2010-06-30,2010-06-30]
AngelaMerkel	type	GermanChancellor	[2005-11-20,now]
AngelaMerkel	type	EnvironmentMinister	[1994-11-17,1998-10-26]
AngelaMerkel	wasBornIn	Hamburg	[1954-07-17,1954-07-17]
AngelaMerkel	wasBornOnDate	1954-07-17	[1954-07-17,1954-07-17]
SaintPeter	wasBornIn	Bethsaida	$null$
FCBayernMunich	won	UEFAChampionsLeague	[2013-05-25,2013-05-25]
FCBayernMunich	won	UEFAChampionsLeague	[2001-05-23,2001-05-23]

Table 2: Examples of SPOT facts

model is the one used by Yago2 [6].

Time is associated with an event or the state of an entity. Events are expressed in a knowledge base through a an event entity (`WorldWarI`), or an instance relation (`wasBornIn`). Instance relations often specify an aspect of an event, such as place and date of the birth event. States are expressed through the `type` relation. Table 2 gives examples of SPOT facts, demonstrating the variations discussed above.

Finally, facts can be associated with textual witnesses such as documents or paragraphs from which they were extracted or where they are mentioned. We call this last dimension of a fact the teXtual dimension (X), which makes a fact in our data model a SPOT+X quint.

3. QUESTION ANSWERING OVER STRUCTURED DATA

Our previous work has tackled the problem of question answering over structured data represented using the SPO triple data model [8]. The goal here is to translate a given natural language question into a structured query that captures the question, and returns a satisfactory answer.

In this work, we tackle questions built from phrases corresponding to entities, classes and relations (collectively known as semantic items). Our task is to disambiguate the question with respect to the data at hand by correctly segmenting the question, mapping segments to the appropriate entities, classes or relations, and grouping them together to capture

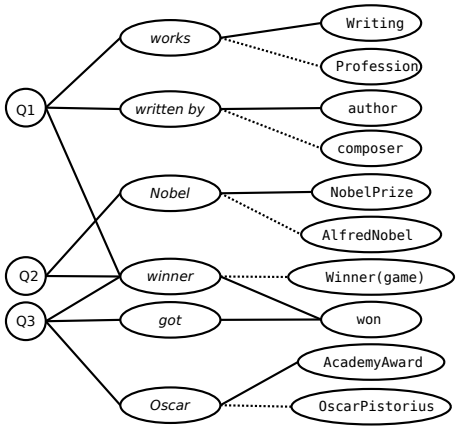

Figure 1: Disambiguation Graph

the structure of the question. As the solution to each of the three disambiguation problems can inform that of the others, we perform these jointly. This joint disambiguation relies on an integer linear program (ILP) that encodes sophisticated constraints that make use of the rich semantics in the underlying data.

Let's consider the question "Which works were written by a Nobel winner who got an Oscar?". Figure 1 shows the *disambiguation graph* that we construct for the question, which encodes all possible interpretations of the question with respect to the knowledge base, and over which the ILP operates. The left edges show dependencies between tokens, which are used to construct SPO triple patterns. Each group of tokens connected to a common Q-node are candidates for being part of the same triple pattern. The right edges map each set of tokens to one or more entity, class or relation. The solid edges are those that are selected by the ILP, while the dotted ones are discarded, corresponding to the SPO triple pattern query:

```
?x type Writing .  ?x author ?y .
?y won AcademyAward .  ?y won NobelPrize
```

which requires properly mapping the ambiguous phrases 'works', 'written by', 'Nobel', 'winner', 'got', and 'Oscar'.

This work, however, did not tackle temporal questions, mainly because these require us to go beyond the triple data model and consider the more expressive SPOT model, that essentially allows for meta facts about basic SPO facts.

4. COPING WITH TIME

We now discuss how our work can be extended to consider the temporal dimension of facts in a knowledge base or other Linked Data sources. The data is assumed to be in the SPOT representation as discussed in Section 2. As before, our goal is to map each question into a structured query, now over SPOT quads rather than just SPO triples.

For generating SPOT quad patterns in a structured query, the key is to identify temporal entities like dates and named events as well as temporal predicates. This is done by (i) extending the space of semantic targets – on the right-hand side of the disambiguation mapping, and by (ii) enhancing the linguistic analysis of user questions with additional temporal cues – on the left-hand side of the disambiguation

mapping. We first discuss the extensions for semantic targets, and address linguistic cues for time-related expressions further below.

Semantic Targets:

To reflect the special nature of time, we introduce a notion of *temporal entities*, or *T-entities* for short. These comprise all explicitly stated timepoints or timespans, such as `24-12-2012`, `Dec-2012`, or `[1960-01-01, 1969-12-31]`, and named events such as `WorldWarII` or `UEFAChampionsLeagueFinal2013`. For explicit dates, we consider different resolutions (e.g. exact date vs. year only) by treating coarse-grained dates as intervals. So `Dec-2012` is equivalent to the timespan `[2012-12-01, 2012-12-31]`. Named events are also associated with explicit timepoints or timespans.

To connect different times of interest, we introduce a notion of *temporal predicates*, *T-predicates* for short. These are binary predicates with two T-entities as arguments. The simplest case is to connect a named event and an explicit date, e.g., `UEFAChampionsLeagueFinal2013 happenedOn 2013-05-25` where `happenedOn` is the T-predicate. Another case is predicates for relative ordering of T-events: `happenedDuring`, `happenedBefore`, `sameYear`, etc. Some of these compare timepoints, others timepoints against timespans, and so on. By using interval notation, we can treat many cases uniformly.

The T-entities that are connected by a T-predicate can be given in the form of an entire SPOT quad such as the data about Angela Merkel's birth `1:AngelaMerkel bornIn Hamburg [1954-17-07]` or her term as Germany's chancellor `2:AngelaMerkel type chancellorOfGermany [2005-11-22, now]`. We would then write `1 happenedBefore 2`, the semantics being that the T-predicate refers to the T dimension of the two connected SPOT facts.

Equipped with these extensions in the semantic target space, we can now represent the SPOT queries that we aim to generate for various questions with temporal aspects.

1: Who was president of the US during the Cuba crisis?
2: Who was president of the US in 1984?
3: Which US president was born in the same year as a Pink Floyd member?

For the example questions above, we would ideally generate the following queries, where T-entities and T-predicates are prefixed with "T:" and the T dimension is always put in brackets [...] (even if it is empty):

```
1:          ?x presidentOf USA [T:?t] .
    T:?t T:overlapsWith T:CubanMissileCrisis [ ]
2:             ?x presidentOf USA [T:?t] .
       T:?t T:overlapsWith [1984-01-01,1984-12-31]
3:   ?x presidentOf USA [ ] .  ?x bornOn T:?t1 [] .
    ?y memberOf PinkFloyd [ ] .  ?y bornOn T:?t2 [] .
             T:?t1 T:sameYear T:?t2 [ ] ] .
```

Linguistic Cues:

The semantic targets are generated from phrases in the question. Dealing with temporal questions requires extending our existing infrastructure to detect cues that may correspond to T-entities and T-predicates.

Detecting temporal expressions that can map to T-entities is achieved using tools such as TimexTag [1] or HeidelTime [7]. Such tools identify spans of tokens of explicit temporal nature, for example, dates. Moreover, these tools often already normalize the temporal expressions. Phrases that refer to named events such as "second world war" or "UEFA final" are detected by our existing infrastructure, drawing from the rich lexical information in large knowledge bases.

Cues for temporal predicates are located using a dictionary of patterns over words and POS tags. These include temporal adverbs (e.g., when), adjectives (e.g., last), conjunctions (e.g., while), prepositions (e.g., before), and specific nouns (e.g., same year as). These surface forms are then translated to T-predicates using a dictionary lookup, and the resulting T-predicates are added to the semantic target space. Note, however, that this step does not yet perform any disambiguation. For example, the preposition "in" can be of temporal or spatial or other nature. Depending on its argument, it may be possible to rule out some options; but in full generality, we often have to generate multiple alternatives. For example, for the phrase "in 1984", we would generate both a year (T-entity) and the book (normal entity) and consequently both happenedInYear (T-predicate) and appearsInBook (normal predicate). The disambiguation mapping (see below) makes the final choice.

By dependency-parsing the question (using the Stanford Parser), we determine dependencies between temporal cues in the question. For each phrase that may denote a T-predicate, we identify its left and right argument in the dependency structure. The arguments (phrases or entire sub-sentences) are are checked as to whether thay may possibly map to T-entities or not. In the latter case, we can prune the candidates for the semantic target space. Also, the dependency structure of the question guides the generation of constraints that will later ensure that T-predicates have the proper kinds of arguments (see below).

Disambiguation Mapping:

With the temporal cues in the input question and the generated candidates for the semantic target space, we can now tackle the mapping from question phrases onto semantic entities, classes, predicates, T-entities, and T-predicates. Once we have a good mapping, we can generate SPOT queries from the mapping. Figure 2 shows disambiguation graphs for two of the example questions given earlier in this section.

As in our earlier work on SPO queries, the key difficulty in this mapping lies in the joint disambiguation of the input phrases. To this end, we extend the ILP model developed in [8]. First we allow T-entities and T-predicates in the range of the mapping and adjust the objective function of the ILP accordingly. Second and specific to our task of coping with temporal questions, we introduce additional constraints that capture the nature of T-predicates. Most importantly, we require that a T-predicate must have T-entities as arguments. This can be in the form of an explicit date or event or by connecting to the T dimension of an entire SPOT fact. In the latter case, we syntactically connect to the P dimension of the SPOT fact but the T-predicate compares the T dimension with the other argument. This is the only way in which T-predicates can be used, and this constraint is encoded in the ILP – first specified as a logical condition over the types of semantic targets, and then cast into a linear condition over the ILP's variables.

As an example, consider question 3, referring to US presidents and Pink Floyd members. Here the T-predicate `sameYear` should connect to two SPOT facts with the `bornOn` predicate for the two persons under comparison; see the SPOT query shown in Figure 2.

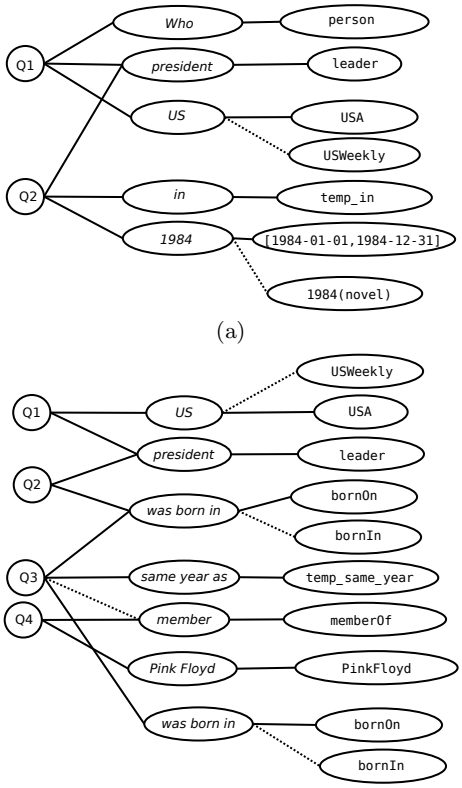

Figure 2: Temporal query disambiguation

5. EXPLOITING TEXT

Sometimes, the temporal aspects of a question may be too sophisticated to be properly mapped into a structured SPOT query. An example could be: "Which songs from the hippie generation years appear in films from this millenium?", where the wording "hippie generation years" may be too difficult for our machinery. In such cases, we can resort to the teXtual extensions of entities and facts, the X dimension, and generate an extended SPOTX query. For example, the above question could be translated into:

```
?s type song [] {"hippie generation years"} .
 ?s playedIn ?m .  ?m type movie [T:?t] {} .
   T:?t T:overlapsWith [2000-01-01,now] {}
```

where `{ }` denotes the X dimension.

Elbassuoni et al. [5] has shown how to evaluate such extended forms of SPARQL queries that combine SPO triple patterns with keyword conditions. In many cases, despite not capturing the temporal aspect of the songs, the SPOTX query can still return informative results, assuming that songs have extensive textual descriptions (e.g., compiled from the full text of Wikipedia or from music consumer portals such as `last.fm`). Such SPOTX queries may also prove useful in cases where the underlying structured data lacks explicit temporal annotations. No knowledge base or database can be complete, so there will often be many songs without and explicit release date. In such cases, the SPOTX query can make up for the missing T data using the X dimension.

SPOTX queries like the one above can be thought of as relaxations of the ideal SPOT query, often producing results of varying informativeness or merely "approximate" results that do not fully reflect the original question's intention. Thus, this approach inevitably calls for result ranking. Berberich et al. [3] have proposed a language model that specifically addresses temporal expressions in text queries and text documents. Our approach builds on this model; we are working on specific extensions.

Some questions are beyond the scope of the relaxed SPOTX query approach. An example is: "Which songs did Leonard Cohen write after Hallelujah?". If the underlying T data has no information on release dates of songs, one would have to answer the temporal parts of the question solely from the text extensions. This is a daunting task, as the texts about songs likely mention many temporal expressions, some of which are release dates or coarse-grained timespans (e.g., "in his mellow years") while others refer to irrelevant aspects (e.g., "used for the Shrek movie in 2001"). Understanding these issues and coping with such complex questions is left for future work.

6. CONCLUSIONS

We introduced the SPOT and SPOT+X models for extending RDF-based Web data into a form that captures the time dimension as well as textual descriptions of entities and facts. We advocate natural-language question answering (QA) as a way for users to search and explore the data. In contrast to mainstream QA methods, our approach translates questions into extended SPARQL queries and harnesses the T and X dimensions of the data. We reported on work in progress on extending our DEANNA system to cope with questions of temporal nature. Given the increasing relevance of temporal information needs, we believe that this line of research is right on the spot.

7. REFERENCES

[1] D. Ahn, J. van Rantwijk, and M. de Rijke. A Cascaded Machine Learning Approach to Interpreting Temporal Expressions. In *HLT-NAACL*, 2007.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives. DBpedia: A Nucleus for a Web of Open Data. In *ISWC/ASWC*, 2007.

[3] K. Berberich, S. J. Bedathur, O. Alonso, and G. Weikum. A Language Modeling Approach for Temporal Information Needs. In *ECIR*, 2010.

[4] K. D. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.

[5] S. Elbassuoni, M. Ramanath, and G. Weikum. RDF Xpress: a flexible expressive RDF search engine. In *SIGIR*, 2012.

[6] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artif. Intell.*, 194, 2013.

[7] J. Strötgen and M. Gertz. Multilingual and Cross-domain Temporal Tagging. *Language Resources and Evaluation*, 2012.

[8] M. Yahya, K. Berberich, S. Elbassuoni, M. Ramanath, V. Tresp, and G. Weikum. Natural Language Questions for the Web of Data. In *EMNLP-CoNLL*, 2012.