TR Discover: A Natural Language Question Answering System for Interlinked Datasets

Dezhao Song[1], Frank Schilder[1], Charese Smiley[1], Chris Brew[2] and Tom Zielund[1], Hiroko Bretz[1], Robert Martin[3], Chris Dale[1], Steven Pomerville[3], John Duprey[3], Tim Miller[4], and Johanna Harrison[5]

[1]Research and Development, Thomson Reuters, Eagan, MN 55123, USA
[2] Research and Development, Thomson Reuters, London, UK
[3] Research and Development, Thomson Reuters, Rochester, NY 14694, USA
[4] Intellectual Property and Science, Thomson Reuters, London, UK
[5] Intellectual Property and Science, Thomson Reuters, Philadelphia, PA 19130, USA
{dezhao.song,frank.schilder,charese.smiley,chris.brew,thomas.zielund,
hiroko.bretz,robertd.martin,chris.dale,steven.pomerville,john.duprey,
tim.miller,johanna.harrison}@thomsonreuters.com

Abstract. We propose TR Discover, a question answering system that answers natural language questions over interlinked datasets. Using a feature-based grammar, TR Discover first parses a natural language question to its First Order Logic representation, which is in turn translated into SPARQL or SQL. Because users will not necessarily know what the coverage of the system is, TR Discover offers a novel auto-suggest mechanism that can help users to construct well-formed and useful natural language questions. We show that TR Discover is usable and portable by applying it to two Thomson Reuters datasets in the Life Science and the Legal domain respectively.

Keywords: Natural Language Interface, Question Answering, Feature-based Grammar, Auto-Suggestion, Analytics

1 Introduction

Non-technical domain experts (e.g., journalists and patent attorneys) can satisfy their information needs through the use of keyword-based search which can be applied uniformly to access information sources with disparate underlying logical and physical structures. However, the result set returned from a keyword-based search may be unwieldy and of limited relevance due to its difficulty in capturing a precise specification of the user's intent. Database query languages, on the other hand, impose structure on the result set and can be used to provide dynamically generated analytics with greater dexterity than less structured results coming from keyword-based search. Still, the learning curve required to command such languages may preclude their widespread adoption by domain experts.

Our system, *TR Discover*, is designed to bridge the gap between keyword-based search and structured query. Using *TR Discover*, the user writes questions

in natural language which are then mapped into a logic-based intermediate language via a feature-based grammar with full formal semantics. Our auto-suggest mechanism steers the user towards logically well-formed questions that are likely to generate useful answers from the available databases. Next, the logical representation of a natural language question is further translated into an executable query (e.g., SPARQL or SQL) thereby allowing the system to use robust existing querying technologies. *TR Discover* enjoys both the advantages of keyword-based search and database query systems by allowing domain experts to use natural language which they already know while retaining precision by mapping from the logical formalism to the query language and generating useful structured analytics. Please refer to our accepted full paper for in-depth technical details [4]. We will demonstrate *TR Discover* using the prototype system available at `http://cortellislabs.com` (freely available after sign-up).

2 System Components

Question Understanding. In *TR Discover*, we parse natural language questions by adopting a feature-based context-free grammar (FCFG). Our FCFG consists of a set of grammar rules that are used to understand the syntactic structure of the questions. The vast majority of these rules are domain-independent, and as such can be re-used when moving to a new domain. As shown below, *G1* - *G3* are a few sample grammar rules. Here, *G3* indicates that a verb phrase (*VP*) may contain a verb (*V*) followed by a noun phrase (*NP*).

G1: NP → (N')
G2: NP→ NP VP
G3: VP → V NP
L1: N[TYPE=patent, NUM=pl, SEM=<λx.patent(x)>] → 'patents'
L2: V[TYPE=[patent,org,file], SEM=<$\lambda X\ x.X(\lambda y$.file_org_patent(y,x))>, NUM=?n] → 'filed by'
L3: V[TYPE=[drug,molecule,target], NUM=?n] → 'targeting'

The lexicon is another component of our FCFG. Each lexical entry contains a variety of domain-specific semantic features which are used to restrict the number of parses that a natural language question may have. In the above example, *L1* represents the lexical entry for *patents*, and specifies its TYPE and semantic information, SEM. Unlike nouns (*L1*), the TYPE of verbs (*L2* and *L3*) specifies both the potential subject-TYPE and object-TYPE, and the predicate name, which helps to filter out nonsensical questions like *patents targeting Anticancer.*

Auto-suggest. Left on their own, users may not know how to begin formulating questions for *TR Discover*. Therefore our system provides suggestions in order to help users to build questions that are likely to be answerable. Unlike Google's auto-completion that is based on query logs, our auto-suggest mechanism provides suggestions computed based upon the entities and their relationships in the dataset and by utilizing the linguistic constraints in our grammar.

As a concrete example, after a user enters "Patents", we could suggest a verb as the next part of the question. In our lexicon, we may have many verbs, such as *filed by*, *granted by*, *developed by*, *utilizing*, etc. Although they all satisfy the grammatical constraints, i.e., they are verbs, only *filed by* and *granted by* are

valid suggestions, since the semantic constraints in our grammar specify that only the subjects of these two verbs can be patents.

FOL Translation and Query Execution. Given a completed natural language question, our system first parses it into a First Order Logic representation (FOL). The FOL of a natural language question is further translated to other executable queries (e.g., SPARQL and SQL). This intermediate logical representation provides us the flexibility to develop different query translators for various types of data stores. There are two sub-steps for translating an FOL to SPARQL/SQL. We first parse the FOL into a parse tree according to an FOL parser, implemented with ANTLR [1]. This FOL parse tree is then translated to executable queries. Finally, the translated queries are executed against their corresponding data stores, i.e., a relational database for SQL queries and a Jena TDB triple store [2] for SPARQL queries.

The following example demonstrates the process of understanding a natural language question and translating it to a SQL and SPARQL query via FOL:

Natural Language Question: `Patents filed by Pfizer`
FOL: `all x.(patent(x) → (file_org_patent(id01,x) & type(id01,Company) & label(id01,Pfizer)))`
SQL Query: `select patent.* from patent where patent.filed_by = 'Pfizer'`
SPARQL Query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX example: <http://www.example.com#>

select ?x
where {
?id01 rdfs:label 'Pfizer'.
?id01 rdf:type example:Company .
?x rdf:type example:Patent .
?x example:filed_by ?id01 .
}
```

Analytics. We provide an overview of the result set with descriptive analytics. For instance, for the question "show me all patents filed by Pfizer", we show the distribution of the countries where Pfizer files their patents. In addition, we perform named entity recognition (using the Stanford CoreNLP toolkit [3]) on the Reuters News Archive (14 million articles), and also conduct sentiment analysis on these news articles. By further linking the recognized companies to those in our database, we show the frequency count of these companies and how their sentiment analysis results change over time. This information may provide further insights to users in order to support their own analyses.

3 Use Case and Demonstration Plan

Figure 1 shows a sample query in progress: *companies developing drugs having a primary indication of . . . ?* As the user types, the system offers possible completions to the question; the user then selects *Cancer*. The pie chart shows each company's market share for cancer drugs. At the bottom of the figure, we display news mentions and sentiment analysis for the most mentioned companies.

Our demonstration of the *TR Discover* system will begin by motivating the use of natural language question answering to uncover information assets found

in diverse, interlinked datasets. We will also illustrate the user's experience of creating questions using guided auto-suggest. Finally, we will explore the resulting analytics and visualizations for various natural language questions highlighting how it allows deeper insights to be gleaned from the data.

We will demonstrate *TR Discover* on two datasets in two different domains: Life Science and Legal. Our Life Science dataset integrates data from different sources: Thomson Reuters drug and patent data, and DrugBank. We interlinked the datasets by matching on certain properties, e.g., company and drug names; thus, relatively comprehensive information can be provided to the users.

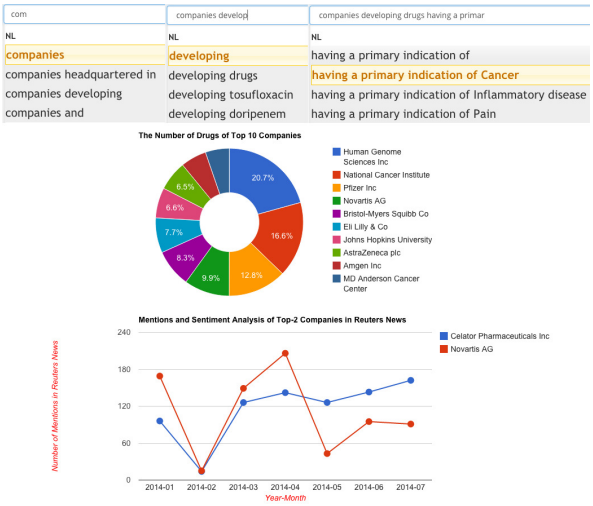

Fig. 1. TR Discover Web User Interface

References

1. Bovet, J., Parr, T.: Antlrworks: an ANTLR grammar development environment. Software: Practice and Experience 38(12), 1305–1332 (2008)
2. Carroll, J.J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: implementing the semantic web recommendations. In: Proceedings of the 13th international conference on World Wide Web - Alternate Track Papers & Posters. pp. 74–83 (2004)
3. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. pp. 55–60 (2014)
4. Song, D., Schilder, F., Smiley, C., Brew, C., Zielund, T., Bretz, H., Martin, R., Dale, C., Duprey, J., Miller, T., Harrison, J.: TR Discover: A natural language question answering system for interlinked datasets. In: The 14th International Semantic Web Conference (2015)