Analyzing and Expanding NPDS Biomedical Record Databases via a Semantic Search Engine

Shiladitya Dutta, Adam G. Craig, and Carl Taswell MD, PhD

***Abstract*—CoVaSEA (Concept-Validating Search Engine Agent) is an automated semantic search engine that is interoperable with the Nexus-PORTAL-DOORS System(NPDS). Utilizing NPDS, CoVaSEA serves to apply SPARQL-based search to external lexical databases by converting relevant biomedical resources to a linked data format which can be searched by the query engine. CoVaSEA consists of 3 components: the SPARQL query engine, the web-crawler, and the quadstore. Acting in concert, these can provide a composite open-web SPARQL search utility. For researchers, this opens the possibility to search external biomedical databases via a semantic web approach and is a solution for situations in which SPARQL queries need to be applied to non-linked data resources.**

***Index Terms*—Nexus-PORTAL-DOORS System, CoVaSEA, SPARQL, web-crawler, semantic web**

Introduction

The Nexus-PORTAL-DOORS System (NPDS)[1] is a system that provides the capability to publish both semantic and lexical resources regarding specific target areas. NPDS has inbuilt REST API services via the Scribe Registrar[2] along with a standardized messaging protocol, enabling exchange among client applications and servers. The structure of the Nexus PORTAL-DOORS System is sub-divided according to the Hierarchically Distributed Mobile Metadata architectural style into three parts: Nexus, PORTAL, and DOORS [3]. The PORTAL registry controls entities that have unique labels and their associated lexical meta-data. The DOORS directory contains semantic meta-data which is primarily comes in the form of RDF descriptions. The Nexus diristry (an aggregation of the terms DIRectory and regISTRY) is a single server that serves as a combination of the PORTAL registry and DOORS directory.

Integrated with NPDS, CoVaSEA (Concept-Validating Search Engine Agent) aims to allows users to apply a SPARQL-query based search to external biomedical literature databases. This approach may be valuable in a variety of tasks which benefit from a combination of a web-crawler's capability to search the open web with the versatility of a semantic query engine. CoVaSEA uses NPDS DOORS directories as a storage basis for the records that it has retrieved, which means that it can also serve as a utility to search and expand NPDS metadata records.

Document created June 20, 2018, revised September 30, 2018; typeset November 4, 2018.

Overview

Acting as a successor to the past CoVaSEA [4], CoVaSEA integrates several features which augment the utility of NPDS including: **(A)** An implementation of SPARQL query based semantic search to allow retrieval and manipulation of linked data descriptions **(B)** Targeted web-crawling for relevant articles from external biomedical literature databases to expand NPDS records **(C)** Automated translation of free-form text abstracts into RDF triples to derive the semantic representations of lexical data. First, the user inputs a Natural Language Query

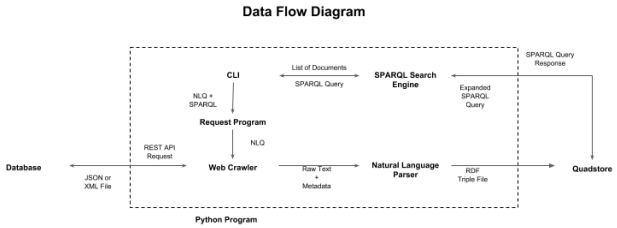

Fig. 1. Data flow diagram of CoVaSEA

(NLQ) and a SPARQL query, which also has the option to be constructed via a SPARQL builder form, into the CLI. These are passed to the webcrawler which combs through the various databases and retrieves the abstract text and metadata for a multitude of articles. On each article, the natural language parser transforms the text and metadata into semantic triples, which are stored in a local quadstore or DOORS directory. These are searched with the provided or constructed SPARQL query and the results are returned to the user. CoVaSEA consists of a number of interdependent components, each of which fulfills a general purpose. The methods are:

- **Web Crawler**: The webcrawler combs through literature databases in order to retrieve relevant articles. Currently the system can search through the literature records on DOAJ, PubMed, Elsevier ScienceDirect, and CORE[5]. Using provided REST API

services, the crawler obtains a list of relevant articles via a general lexical search. From there, the basic meta-data, consisting of the abstract, title, author(s), database of origin, DOI(if available), and date of publication, from each article is retrieved and passed to the natural language parser. An RDF representation of the abstract will be constructed and the rest of the meta-data will be included in the named graph of the article via the DublinCore ontology[6].

- **Natural Language Parser**: The Natural Language Parser receives the raw abstract text from the web crawler and parses it into logical form triples. This section utilizes the Stanford Core NLP library[7] and NLTK [8].

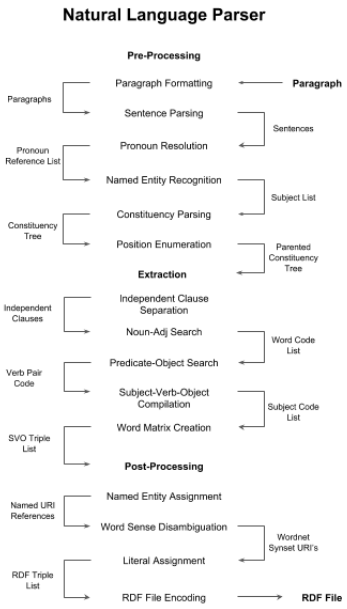

Fig. 2. Data flow diagram of the Natural Language to RDF parser

  - Pre-processing: A step in which the raw text is prepared for parsing and to perform tasks that depend on the text itself such as Named Entity Recognition[9] and Co-Referencing[10]. Preprocessing preparation includes removing troublesome characters, re-spacing the text to ensure homogeneity, and parsing the paragraphs into sentences.
  - Extraction: Independent clauses are separated from each other and a constituency parse occurs on each independent clause. Breadth-first and depth-first search are used to find subject and verb-object phrases. Breadth-first search is used to find the highest noun phrase in the tree. This noun phrase is split up if there are conjunctions and any adjectives are linked to the nouns. Then a two stage breadth-first and depth-first search is used to find the verb. Breadth-first search is used to find the highest verb phrase and depth-first search is used to separate the verb from the prepositional or adverb sections. The verbs and noun phrases are then tagged with positions and put into triples.
  - Post-processing: Graph Compression is performed and the logical form triples are translated into RDF triples. Unique Resource Identifiers(URI) are assigned to each part of the triples. First field-specific language and some named entities are assigned URI's via databases such as MeSH. Then standard words are assigned via WordNet[11] synsets based on context and part of speech using Lesk[12]. Finally named entities and numbers are assigned to literals. These steps aid in graph-compression and also allows for the full semantic meaning to be captured rather than just the lexical data. The triples are all packaged into the article's named graph.

- **SPARQL Analysis Engine**: A SPARQL query is received and then used to search through the triple store to extract relevant data for the user. For a more conducive user experience, the query engine supports a SPARQL builder, thus circumventing the need to know SPARQL syntax. The system works by having the user create a series of either filtering, mandatory, or optional conditions in either an independent or nested connection. Thus the user can build a search query via simple fill-in-the-blank statements. Though it cannot replicate the full power of SPARQL syntax, it is a potent resource for users do not want to use SPARQL. Due to the dynamic and heterogeneous nature of the data that CoVaSEA handles, the SPARQL query engine divides up the queries into sub-queries such that only the articles that are needed are compiled. This federated approach avoids the necessity of having to compile a computationally expensive local graph. Since the system is based off of named graphs, it is relatively easy to trace the article of origin of a particular triple if necessary. This iteration based approach helps to optimize runtime and memory-load on computers by eliminating the redundancy of loading unneeded graphs.
- **QuadStore**: To prevent redundant re-rendering of articles, CoVaSEA records the triples which are built by the web-crawler in either a local quadstore or a DOORS directory. Each of the articles in the record has its own named graph, allowing for the identification of the origin of triples. The graph is divided into two sections: the meta-data section and the semantic representation section. The meta-data section stores key information about the article such as author, title, publication date, and database of origin. The semantic representation section stores a set of triples which are a RDF portrayal of the article's

abstract. The RDF files are sent and retrieved from the DOORS directory utilizing the Scribe API.

Results

CoVaSEA currently has a run time for a full search of 245.12 seconds for 20 articles and 1307.42 seconds for a 100 articles. The breakdown of the runtime for 20 articles is 102.32 seconds for the web-crawler to retrieve 20 articles from the 4 databases, 139.47 seconds to parse the abstracts into RDF files, and 3.21 seconds for the SPARQL query search. The breakdown for a 100 articles is 483.13 seconds for the web-crawler, 721.47 seconds for parser, and 102.82 seconds for the SPARQL query search. Note that all tests are run on a computer with 16 GB of RAM and a i7-8750H. The connection speeds are a download of 18.40 Mgbs and an upload of 1.68 Mgbs. It should be noted that most of the articles retrieved had relatively short abstracts of 1 paragraph.

Discussion

CoVaSEA offers capabilities that can facilitate a variety of tasks. Primarily, the role of CoVaSEA is a utility to apply a SPARQL-query based search to external biomedical literature databases. For researchers, this may be useful in situations for which the versatility and adaptability that is offered by a SPARQL query needs to be applied to lexical data on the open-web. However, CoVaSEA is not just limited to serving the user, but also, due to its relationship with NPDS, is a utility to search and expand semantic records on the DOORS directory. Since system is automated, and can furnish large amounts of semantic descriptions on a regular basis, CoVaSEA lays the groundwork for a variety of future NPDS applications(i.e. automated meta-analysis).

The primary point of possible expansion for CoVaSEA is the Natural-Language Parser. Currently, the parser faces difficulty in 3 main areas: relevance of triples, accuracy of triples, and runtime. The first problem is the relevance of triples and how well they summarize the passages. The second issue is the accurate parsing of advanced sentence structures. The final problem is that of optimization, which impedes us from performing whole articles parses with reasonable run-times. Another section of improvement is the SPARQL query engine. The two biggest improvements that could be made are SPARQL query expansion and a semantic reasoning engine, both of which increase the breadth of the search.

Conclusion

Here we presented CoVaSEA: a system in which resources from the open-web can be translated into machine-understandable semantic information and be searched via SPARQL. CoVaSEA has the capability to both search externally with the web crawler for semantic data to expand the NPDS knowledge base and internally with SPARQL to navigate the data inside the DOORS directory. These capabilities combine to offer a utility which can semantically search external biomedical literature databases. The application of the wide range of possibilities offered by semantic search to the array of lexical information present on the open-web renders many services, both for users and for the Nexus PORTAL-DOORS System.

References

[1] C. Taswell, G. TeleGenetics, and C. Ladera Ranch, "Portal-doors infrastructure system for translational biomedical informatics on the semantic web and grid," *Proceedings of the American Medical Informatics Association Summit on Translational Bioinformatics, San Francisco, CA*, p. 43, 2008.

[2] A. Craig, S. H. Bae, T. Veeramacheneni, *et al.*, "Web service apis for scribe registrars, nexus diristries, portal registries and doors directories in the npd system.," in *SWAT4LS*, 2016.

[3] C. Taswell, "A distributed infrastructure for metadata about metadata: The hdmm architectural style and portal-doors system," *Future Internet*, vol. 2, no. 2, pp. 156–189, 2010.

[4] S.-H. Bae, A. G. Craig, C. Taswell, *et al.*, "Expanding nexus diristries of dementia literature with the npds concept-validating search engine agent,"

[5] P. Knoth and Z. Zdrahal, "Core: Three access levels to underpin open access," *D-Lib Magazine*, vol. 18, no. 11/12, 2012.

[6] S. Weibel, "The dublin core: A simple content description model for electronic resources," *Bulletin of the American Society for Information Science and Technology*, vol. 24, no. 1, pp. 9–11, 1997.

[7] C. Manning, M. Surdeanu, J. Bauer, *et al.*, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.

[8] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[9] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the 43rd annual meeting on association for computational linguistics*, Association for Computational Linguistics, 2005, pp. 363–370.

[10] K. Clark and C. D. Manning, "Entity-centric coreference resolution with model stacking," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, 2015, pp. 1405–1415.

[11] G. A. Miller, "Wordnet:a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[12] S. Banerjee and T. Pedersen, "An adapted lesk algorithm for word sense disambiguation using wordnet," in *International conference on intelligent text processing and computational linguistics*, Springer, 2002, pp. 136–145.