Open Data Search Framework based on Semi-structured Query Patterns

Marut Buranarach[1], Chonlatan Treesirinetr[2], Pattama Krataithong[1] and Somchoke Ruengittinun[2]

[1] Language and Semantic Technology Laboratory
National Electronics and Computer Technology Center (NECTEC), Thailand
{marut.bur, pattama.kra}@nectec.or.th

[2] Department of Computer Science, Faculty of Science, Kasetsart University, Bangkok, Thailand
somchoke.r@gmail.com

Abstract. Open government data (OGD) is a global initiative to promote transparency, service innovation and citizen participation. OGD is usually made available in forms of datasets on OGD web portals. Searching OGD is usually conducted using metadata search on OGD catalogs. Although searching OGD based on metadata or full-text search is common, it cannot take full advantage of the structured data content in the datasets. By being able to query data in the datasets, the user can find the relevant information more effectively. This paper proposes an open data search framework based on semi-structured query patterns. The proposed semi-structured query pattern has more structured than typical keyword search which will allow for more expressive query. It is also less rigid than structured query which reduces the user effort in forming a query. Three query patterns are currently supported and can be converted to API requests to the existing dataset APIs of Data.go.th. The query suggestion module of the system can make suggestions for possible queries based on the user's initial typing. A prototype system was created to demonstrate searching some datasets from Data.go.th using this approach. Finally, we discuss some lessons learned and current limitations that should be improved in future work.

Keywords: open data search, semi-structured question, dataset API

1 Introduction

Open government data (OGD) is a global initiative to promote transparency, service innovation and citizen participation. The most common means for publishing OGD is usually in forms of datasets made available on OGD portals such as *Data.gov*, *Data.gov.uk* and many others. Searching OGD datasets usually relies on search functions of OGD portal software such as CKAN[1] in searching their data catalogs. The search functions are usually based on keyword-based search over metadata fields or tag-based search. Although searching datasets based on metadata is straightforward and

[1] https://ckan.org/

can help the user to find relevant datasets, the user needs to look into each dataset to find the information he or she is looking for in each dataset. For example, if the user is looking for a phone number of a school, the user may have to search for the datasets whose metadata contains the term "school" and then looks into each returned dataset whether it contains the telephone number information. Even when full-text indexing and searching is applied, the user may only find the datasets containing the search terms but not the "answer" the user is looking for. Effective mechanism that can allow for "data-level" querying in addition to "dataset-level" querying is needed for querying OGD datasets.

There are typically two main approaches in querying structured data: keyword-based and structured query. Using keyword-based query, the search system searches the data on every fields. Thus, the structure information of the data is not used in the query. This approach has an advantage that it reduces user effort in forming a query with a disadvantage of limited query expressiveness. Using structured query, which is typically specified via form-based interface, the search system transforms the user query to a structured query language expression, i.e. SQL, in searching the data. This approach has an advantage that the user can specify expressive query with a disadvantage of requiring more user effort in forming query.

In this paper, we propose a semi-structured query approach based on query patterns as an additional form of querying OGD datasets. In this approach, user can specify search conditions in free-text from with auto-complete suggestions for the possible query terms and conditions based on some defined query patterns. For example, the user can define a query such as "rajini school telephone" to search for the telephone number of the school. Currently, three query patterns are defined. The search system utilizes dataset APIs created for some datasets on Data.go.th [1]. The APIs were provided on top of an RDF database. Specifically, the OGD datasets were converted to the RDF data format. The query patterns were mapped with some pre-defined API and SPARQL query templates. We developed a prototype system for searching some OGD datasets from Data.go.th using this approach. Finally, some potentials and limitations of the framework are discussed.

2 Related Work

Our approach relies on RDF data querying using SPARQL query templates. We briefly review some related work on linked open data search focusing on querying interface as follows. RDF Xpress [2] provides a form-based search interface for searching linked data sources. The user can combine triple patterns with keywords to form queries with auto-complete feature. This work also defines the following components for linked data search system: RDF knowledge base, search interface, retrieval engine, query relaxer and result diversifier. [3] discussed some unique challenges for linked data search engine including the user interface issue. [4] investigated a natural language query mechanism for linked data by mapping user queries into some query graph patterns. To the best of our knowledge, our work is the first that proposes a generic framework for querying OGD datasets based on data-level querying using semi-structured query patterns.

3 An Open Data Search Framework based on Semi-structured Query Patterns

3.1 Conceptual Architecture

A conceptual architecture of the open data search framework based on semi-structured query patterns is shown in Fig. 1. The system consists of four major modules: Dataset APIs, Query Translation, Query Suggestion and Result Formatter. Each module is briefly described as follows.

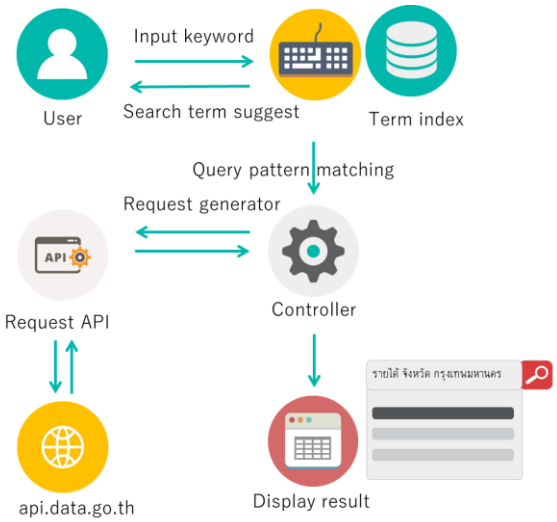


Fig. 1 A conceptual architecture of the open data search framework based on semi-structured query patterns

Dataset APIs: Publishing RDF and data API from existing OGD datasets can further promote application and integration of OGD. Our previous work has proposed a semi-automatic mechanism for such a process [1]. The data publishing and querying system was extended from the OAM framework [5]. Some datasets from Data.go.th have been transformed and published as RDF datasets, i.e. via direct mapping, and RESTFul APIs. The API requests were translated into SPARQL queries based on predefined query patterns. The returned results were formatted to the JSON format.

Query Translation: In our framework, three semi-structured query patterns were defined. The user can post a query in one of the patterns. The query patterns were subsequently translated into API requests made to the available dataset APIs. If the query is not in the defined patterns, the query is treated as typical keyword search.

Query Suggestion: In our framework, a semi-structured query pattern is defined as a query that does not have a rigid structure but having a more controlled form than keyword search. Thus, in order to prevent the user from forming the malformed query, a query suggestion module was developed. The module relied on a created index

of the relations between property, class and values from the data in the datasets. It suggests possible classes, properties and values based on the user's initial typing for the query.

Result Formatter: The results from the dataset APIs in the JSON format were transformed into a table format. Although the results were presented in table form, the likely answer is also highlighted within the table cells.

3.2 Query Patterns and API Request Translation

In our framework, three semi-structured query patterns were defined. The user can post a query in one of the following patterns in the triple format.

Pattern 1: <class> <property> <value>
Pattern 2: <property> <subject>
Pattern 3: <subject> <property>

In Pattern 1, the objective is to retrieve the instances of a class that matched with the query condition <property> = <value>. For example, a query "income province bangkok" will retrieve instances of the class 'income' whose 'province' property has the value 'bangkok'. A specified class name must be mapped with dataset tags and resolved to some targeted datasets. Then a query is formed and run against the datasets. The follows is an example API request for such a query.

query?dsname=income&path=income&property=province&operator=CONTAINS&value =bangkok

In Pattern 2, the objective is to retrieve the value of a given property of a given instance. For example, a query "telephoneNo Rajini School" will retrieve the instance of 'Rajini school' and highlighted the value of the 'telephoneNo' property in the result. In this pattern, the instance and property terms must be checked for the datasets that contain the terms. A query to search the data related to this instance was then run against the datasets. The results were highlighted for the value of the given property. The follows is an example API request for such a query.

query?dsname=school&path=school&keywords=rajini%20school

Pattern 3 is similar to Pattern 2 except that the positions of the subject and property terms are switched. The API translation is the same as that of Pattern 2.

3.3 Query Suggestions

The system makes suggestions to the user for possible queries given the user initial characters for the query. In order to make suggestions, the possible classes (dataset tags), properties and values must be collected and indexed from the text data in the datasets. An ER diagram showing entities and relationships of terms for making query suggestions is shown in Fig. 2. The diagram presents a ternary relationship between dataset or class, property and value terms. Given this database design, the listing and

possible relationships between datasets, properties and value terms can be retrieved from the database. The value terms only include string values within a given length limit. This allows the auto-complete function to be applied when the user is typing characters and terms. A resulted query made by the auto-complete function will result in a valid query made to the API.

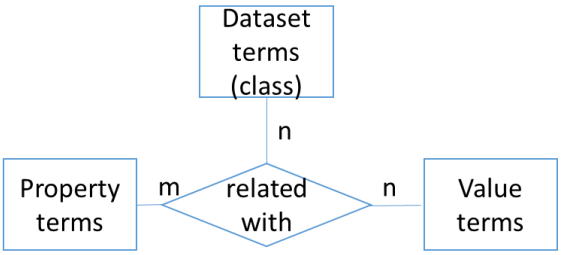


Fig. 2 An ER diagram showing entities and relationships of terms for making query suggestions

4 Case Study

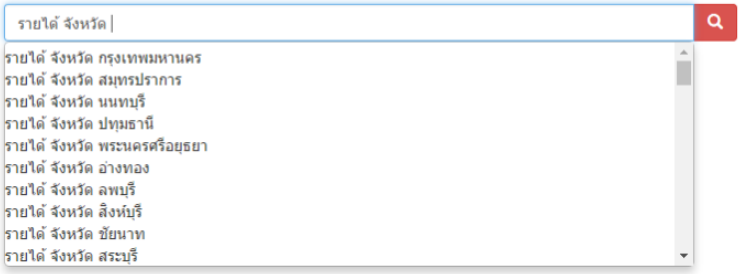
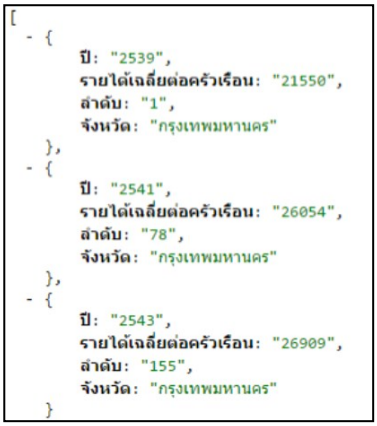
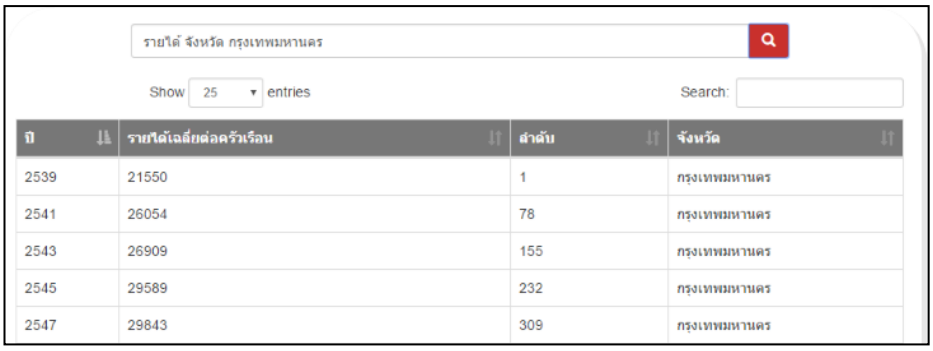


Fig. 3 An example query suggestions for the query pattern 1 “income province bangkok”

A prototype system was developed using about ten datasets from Data.go.th to demonstrate the framework. Dataset APIs were created for these datasets. The terms in these datasets were indexed for the query suggestions module. The total number of the indexed properties and term relations were over 160 and 25,000 entries respectively. Fig. 3 shows an example query suggestions for the query pattern 1. In this example, the user initially types “income” and the suggested terms are the list of possible property for this class (dataset). Once a property is selected, e.g. “income province”, the list of possible values, which are province names, in the dataset is suggested. The user can select a value, e.g. “income province bangkok”. The system then converted the query to an API request to query the dataset API with the given criteria. Fig 4a and 4b shows the query result in both JSON and table formats.



a) Example query results from the income statistics dataset API in JSON format



b) Example query results of the system in table format

Fig. 4 Example result listing yearly income statistics of Bangkok in the JSON and table formats

5 Conclusion and Discussion

This paper proposes a framework for searching data in OGD datasets. The framework allows the user to post semi-structured query patterns in querying the data in the OGD datasets. The proposed semi-structured query pattern has more structured than typical keyword search which will allow for more expressive query. It is also less rigid than structured query which reduces the user effort in forming a query. The result is similar to the result of database querying. Three query patterns are currently supported and can be converted to API requests to the existing dataset APIs of Data.go.th. The query suggestion module of the system can make suggestions for possible queries based on the user's initial typing. The module requires indexing of terms and their relationships

in the datasets in terms of classes, property and values. A preliminary prototype system was created to demonstrate searching a small number of datasets from Data.go.th using this approach.

Based on our prototype system, we discuss some lessons learned as follows. Although the system can work well with a small number of datasets, it is currently not highly scalable. With the increasing number of datasets, the number of the indexed terms and their relations is rapidly grows. This can greatly reduce the performance of the system in making query suggestion. In the future, the index may be created in NoSQL database to improve its scalability. In addition, more supported query patterns should be provided. For example, a query pattern which consists of multiple query conditions, e.g. "income province bangkok year 2015", should be additionally provided. Currently, the property terms relied on the terms used in the column headers. However, some header labels in the datasets are ambiguous or not meaningful, e.g. 'TelNo' label to represent telephone number. This can result in some query suggestions that are ambiguous or not meaningful. Future work should focus on these issues to improve the performance and usability of the framework.

Acknowledgment

This project was partially supported by the Electronic Government Agency (EGA) and the National Science and Technology Development Agency (NSTDA), Thailand.

References

1. Krataithong, P., Buranarach, M., Supnithi, T.: RDF Dataset Management Framework for Data.go.th. In: Proceedings of the 10th International Conference on Knowledge, Information and Creativity Support Systems (KICSS2015) (2015).
2. Elbassuoni, S., Ramanath, M., Weikum, G.: RDF Xpress: A Flexible Expressive RDF Search Engine. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 1013. ACM, New York, NY, USA (2012).
3. Hogan, A., Harth, A., Umbrich, J., Kinsella, S., Polleres, A., Decker, S.: Searching and Browsing Linked Data with SWSE: The Semantic Web Search Engine. Web Semant. 9, 365–401 (2011).
4. Freitas, A., Oliveira, J.G., O'Riain, S., da Silva, J.C.P., Curry, E.: Querying linked data graphs using semantic relatedness: A vocabulary independent approach. Data Knowl. Eng. 88, 126–141 (2013).
5. Buranarach, M., Supnithi, T., Thein, Y.M., Ruangrajitpakorn, T., Rattanasawad, T., Wongpatikaseree, K., Lim, A.O., Tan, Y., Assawamakin, A.: OAM: An Ontology Application Management Framework for Simplifying Ontology-Based Semantic Web Application Development. Int. J. Softw. Eng. Knowl. Eng. 26, 115–145 (2016).