FedViz: A Visual Interface for SPARQL Queries Formulation and Execution

Syeda Sana e Zainab[1], Muhammad Saleem[2], Qaiser Mehmood[1], Durre Zehra[1], Stefan Decker[1], and Ali Hasnain[1]

[1] Insight Centre for Data Analytics, National University of Ireland, Galway
firstname.lastname@insight-centre.org
[2] Universität Leipzig, IFI/AKSW, PO 100920, D-04009 Leipzig
{lastname}@informatik.uni-leipzig.de

Abstract. Health care and life sciences research heavily relies on the ability to search, discover, formulate and correlate data from distinct sources. Over the last decade the deluge of health care life science data and the standardisation of linked data technologies resulted in publishing datasets of great importance. This emerged as an opportunity to explore new ways of bio-medical discovery through standardised interfaces. Although the Semantic Web and Linked Data technologies help in dealing with data integration problem there remains a barrier adopting these for non-technical research audiences. In this paper we present FedViz, a visual interface for SPARQL query formulation and execution. FedViz is explicitly designed to increase intuitive data interaction from distributed sources and facilitates federated as well as non-federated SPARQL queries formulation. FedViz uses FedX for query execution and results retrieval. We also evaluate the usability of our system by using the standard system usability scale as well as a custom questionnaire, particularly designed to test the usability of the FedViz interface. Our overall usability score of **74.16**% suggests that FedViz interface is easy to learn, consistent, and adequate for frequent use.

Keywords: SPARQL, Life Sciences (LS), Query Federation, Visual Query Formulation

1 Introduction

The researchers in health care, life sciences and biomedical (also known as domain users) adopted Semantic Web and Linked Data technologies due to the data integration challenges faced as a result of excessive data produced [6,16]. Different researchers recommended the use of SPARQL services for publishing biomedical resources [2,20,19]. The use of these technologies facilitate the domain users for issuing structured SPARQL queries over highly heterogeneous data spread over diverse data sources [5,1]. Such structured queries are vital, not only in order to query relevant data regarding different entities e.g. Drugs, Molecules and Pathways but also to drive meaningful biomedical correlations such as Drug Drug Interactions and Protein Protein Interactions etc. Such retrieved information can subsequently be applied to various bioinformatics tasks such as functional analysis, protein modelling or image analysis. As pointed out earlier that

in the most of cases, the required information to draw any biological correlation or to answer a biological question involve querying multiple data source, provided by different providers, sometimes available in different format with different accessing mechanism. Meaningful biological query such as *"Find out the Diseases that causes due to the deficiency of Iodine"* can only be answered by querying and aggregating data from multiple reliable data sources. The use of Semantic Web and Linked Data technologies are commonly exploited by computer scientists, who can formulate structured SPARQL queries to access data from different SPARQL endpoints, the ultimate end-users and the domain experts either biologists or clinical researchers, remain unable to assemble complex queries in order to access such data [8]. Making complex SPARQL queries to drive necessary information to support clinical experiments and observations poses a barrier in health care and life sciences domain that confront the adoption and acceptance of such technologies. Moreover, even for computer scientists, assembling a federated SPARQL query is time-consuming and technical process since it requires the knowledge of underlying datasets schema and the connectivity between the datasets [9,10]. An alternative to this is an intuitive and interactive platform that can facilitate domain users to assemble complex but meaningful SPARQL query through visual interface. To this end, we introduce FedViz which enables a user to formulate and execute complex federated SPARQL queries using intuitive visual query interface. FedViz allows user to select concepts and properties from multiple datasets using nodes and edges, assemble SPARQL query in a background independent of user involvement and allow users to edit the resultant SPARQL query before sending it to the SPARQL query federated engine. Assembled query is executed through FedX- a state of the art engine [22], that federates the query to relevant data sources and retrieves the results. The choice of FedX was due to the fact it can execute both federated (both SPARQL 1.0 and SPARQL 1.1) and non federated queries. At present, six real time biomedical data sources, i.e., Kegg, Drugbank, DailyMed, Medicare, Sider, and Diseasome are selected to visually construct the SPARQL query. However, FedViz can be generalise to any set of datasets.

The remaining part of this paper is organised as follows: we highlight the related work in section 2. Later we present the motivational use case in section 3. We introduce our methodology and FedViz salient features in section 4. Subsequently, we present a thorough evaluation of FedViz in section 5. We finally conclude the paper with an overview of future work.

2 Related work

Several approaches have been proposed for Visual query formulation over Linked data. *Form-based querying* is one of the famous paradigm, where *Form elements* (i.e. filters, variables, identifiers) are used for query formulation. Example of this approach is SPARQLViz [3]. However it is less flexible and allows only those users with some knowledge of RDF and SPARQL language. In *Graph-based querying* paradigm query is formulated using node-link diagrams and this approach is more flexible as compared to *Form-based paradigm* and requires the RDF notations of subject-predicate-object cause barrier for users with limited semantic web knowledge. Examples for such approaches include NITELIGHT [15], iSPARQL[1], RDF-GL [11] and ReVeaLD [13]. QueryVOWL[7] uses

[1] http://oat.openlinksw.com/isparql/

Listing 1.1: Find all the drugs and their interactions for curing thyroid disease.

```
PREFIX drugbank: <http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugbank/>
PREFIX diseasome:<http://www4.wiwiss.fu-berlin.de/diseasome/resource/diseasome/>
Select Distinct ?interactionDrug1 ?interactionDrug2 ?text ?name
WHERE
{
?Drugbank0 a drugbank:drug_interactions;
drugbank:interactionDrug1 ?interactionDrug1;
drugbank:interactionDrug2 ?interactionDrug2;
drugbank:text ?text.
?interactionDrug1 drugbank:possibleDiseaseTarget ?possibleDiseaseTarget.
?possibleDiseaseTarget diseasome:name ?name.
FILTER (regex(?name, "thyroid" , "i" ) )
}
LIMIT 100
```

specific language and graph database. Most of aforementioned available systems focused on query formulation using specific graphs, available predicate links and user may need sufficient SPARQL knowledge using such system. FedViz is a step towards interactively and intuitively formulating federated SPARQL queries using class and property links visually presented per dataset.

3 Motivation

We believe FedViz enables a variety of use cases, of which one is explained as follows: **Drug-Drug Interaction for Medication of Certain Disease:** When patients are diagnosed with certain disease, a large number of drugs are associated with that depending upon its stage and condition. It is imperative that physician are thoroughly educated about drug-drug interaction before prescription for certain disease. Take `hypothyroidism` for example. It is a disease which results from an under-active thyroid, leading to the necessity of taking extrinsic thyroxine hormone to maintain normal bodily functions. One treatment option for hypothyroidism is using `Levothyroxine`, which is a synthetic thyroid hormone similar to T4 hormone, which is intrinsically produced by the thyroid gland, deficiency of which leads to the disease in the first place. `Levothyroxine` has many drug interactions, especially with the warfarin family and similar drugs, including `Acenocoumarol`. It is an anticoagulant that functions as a Vitamin K antagonist, and so controls clot formation in the body. Simultaneous use of `Levothyroxine` with `Acenocoumarol` can sensitise the body to the latter, which may put the patient at an increased risk of bleeding. This is just an example how FedViz can be used to monitor interactions of a drug, in this particular case `Levothyroxine`, by creating a visual query, making it easier for the physician to have a comprehensive look at the potential contraindications to using the drug in particular patients (Listing 1.1).

4 FedViz

FedViz is an online application that provides Biologist a flexible visual interface to formulate and execute both federated and non-federated SPARQL queries. It translates the visually assembled queries into SPARQL equivalent and execute using query engine.

At present, FedViz visualises Life Sciences datasets and facilitates complex query formulation and execution in order to draw meaningful biological co-relations including drug-drug interaction, drug-disease interaction and drug-side effect correlations. Through FedViz Biologist can formulate simple queries that typically involve single or multiple concepts from one dataset as well as complex federated queries that might involve more than one datasets with multiple constraints.

4.1 Methodology

Our methodology consists of two steps namely: 1) building visual interface and 2) result retrieval using query engine (Figure 1).

Building visual interface A concise graphical representation is needed to display datasets to facilitate biologist in order to formulate query. We chose the concept map approach [12] for building the visual interface, which is a graphical method representing the relationship between nodes and links, and has been used in various domains for organising knowledge [24]. Using this approach in FedViz, we represent concepts as big circular nodes (drugs, disease etc) and properties as small circular nodes (protein sequence, possible disease target etc). As mentioned earlier, currently FedViz contains six datasets and their concepts with associated properties are visualised for query formulation also known as catalogue (Fig 1). Each dataset represented in catalogue is marked with unique colour. The nodes are modelled as objects in a two-dimensional system using a force-directed layout[23]. In force-directed layout nodes repel each other based on their sizes that prevents overlapping and increases concept-property visibility to end-user.

Result Retrieval Using Query Engine To process the FedViz query request, FedX the state of the art efficient SPARQL query federation engine [18] is chosen to execute both federated (SPARQL 1.1 and SPARQL 1.0) and non-federated queries. FedViz provides the set of required SPARQL endpoints (i.e., data sources) URLs in order to enable FedX's query execution. Overall, the query execution works as follow: (1) FedViz formulate SPARQL query and sends to FedX, (2) FedX executes the query and sends back the results to FedViz, (3) FedViz presents the results to end user.

Technologies FedViz is browser-based client application that provides biologist a flexible front-end. To build this application variety of web technologies are used including HTML5, CSS, JavaScript, JQuery[2], Java Servlet, SVG[3], AJAX[4] and JSON[5]. The datasets visualisation is based on SVG (Scaler Vector Graphics) with Javascript usage. In catalogue, datasets are represented in JSON format and displayed as nodes (Concept and Properties). The communication between the client query and federated query engine(FedX) has done by AJAX calls through middle layer. Open source Javascript library D3.js[4] is used to implement force-directed layout for datasets visualisation.

[2] https://jquery.com/
[3] www.w3schools.com/svg/
[4] http://api.jquery.com/jquery.ajax/
[5] http://json.org/

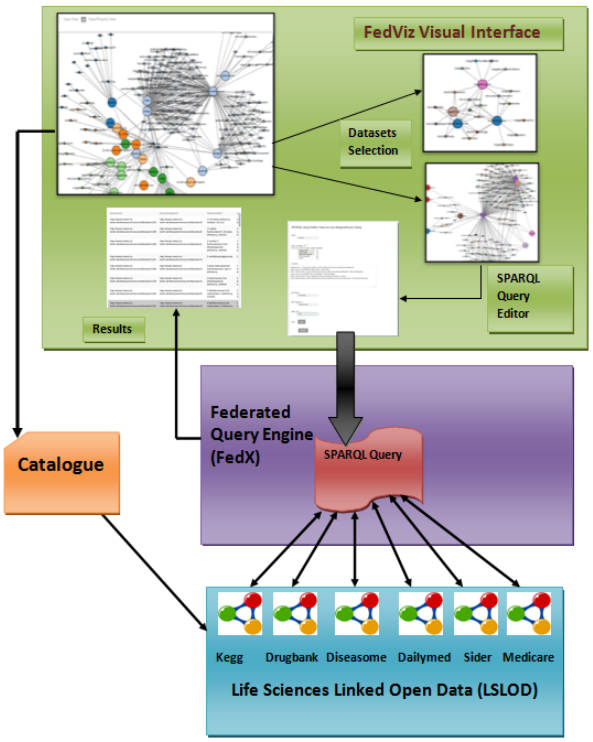

Fig. 1: FedViz Architecture Diagram

Availability The FedViz application can be accessed at http://srvgal86.deri.ie/FedViz/index.html. Example queries both simple (include single dataset) and complex (include more than single dataset) are provided at https://goo.gl/AOJGpu.

4.2 Datasets

Current version of FedViz supports a total of 6 real-world datasets. All the datasets were collected from Life Sciences domains. We began by selecting two real world datasets from Fedbench [21] namely Drugbank[6] a knowledge base containing information of drugs, their composition and their interactions with other drugs and Kegg Kyoto Encyclopedia of Genes and Genomes (KEGG)[7] which contains further information about chemical compounds and reactions with a focus on information relevant for geneticists. Apart from aforementioned selected datasets four other datasets were chosen that had connectivity with the existing ones that enabled us to include real federated queries. These datasets include Sider[8]- that contains information on marketed drugs and their

[6] http://www.drugbank.ca/
[7] http://www.genome.jp/kegg/
[8] http://wifo5-03.informatik.uni-mannheim.de/sider/

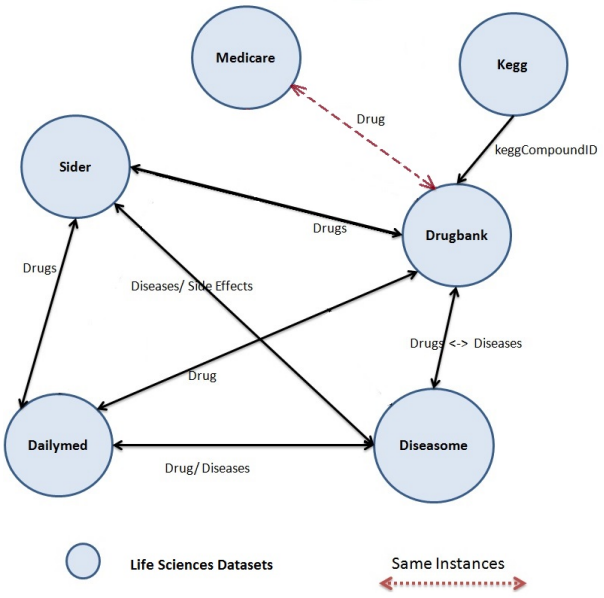

Fig. 2: Datasets Connectivity.

adverse effects, Diseasome[9]- that publishes a network of 4,300 disorders and disease genes linked by known disorder-gene associations for exploring all known phenotype and disease gene associations, indicating the common genetic origin of many diseases., Dailymed[10]- provides information about marketed drugs including the chemical structure of the compound, its therapeutic purpose, its clinical pharmacology, warnings, precautions, adverse reactions, over dosage etc., and Medicare[11]. Figure 2, shows the topology of all 6 datasets while some other basic statistics like the total number of triples, the number of resources, predicates and objects, as well as the number of classes and the number of links can be found in table 1.

4.3 Query Formulation

In this section, an example scenario is discussed to demonstrate our visual query formulation process.

Drug-Disease and Drug-Compound interaction: *Drugs with their compound mass for curing disease Anemia.* This query requires data integration from Drugbank (containing drugs information), Diseasome (containing disease information) and Kegg(containing compound mass information) and can be formulated by using the following step-by-step approach (ref., Fig. 3):

[9] http://wifo5-03.informatik.uni-mannheim.de/diseasome/
[10] http://dailymed.nlm.nih.gov/dailymed/index.cfm
[11] http://wifo5-03.informatik.uni-mannheim.de/medicare/

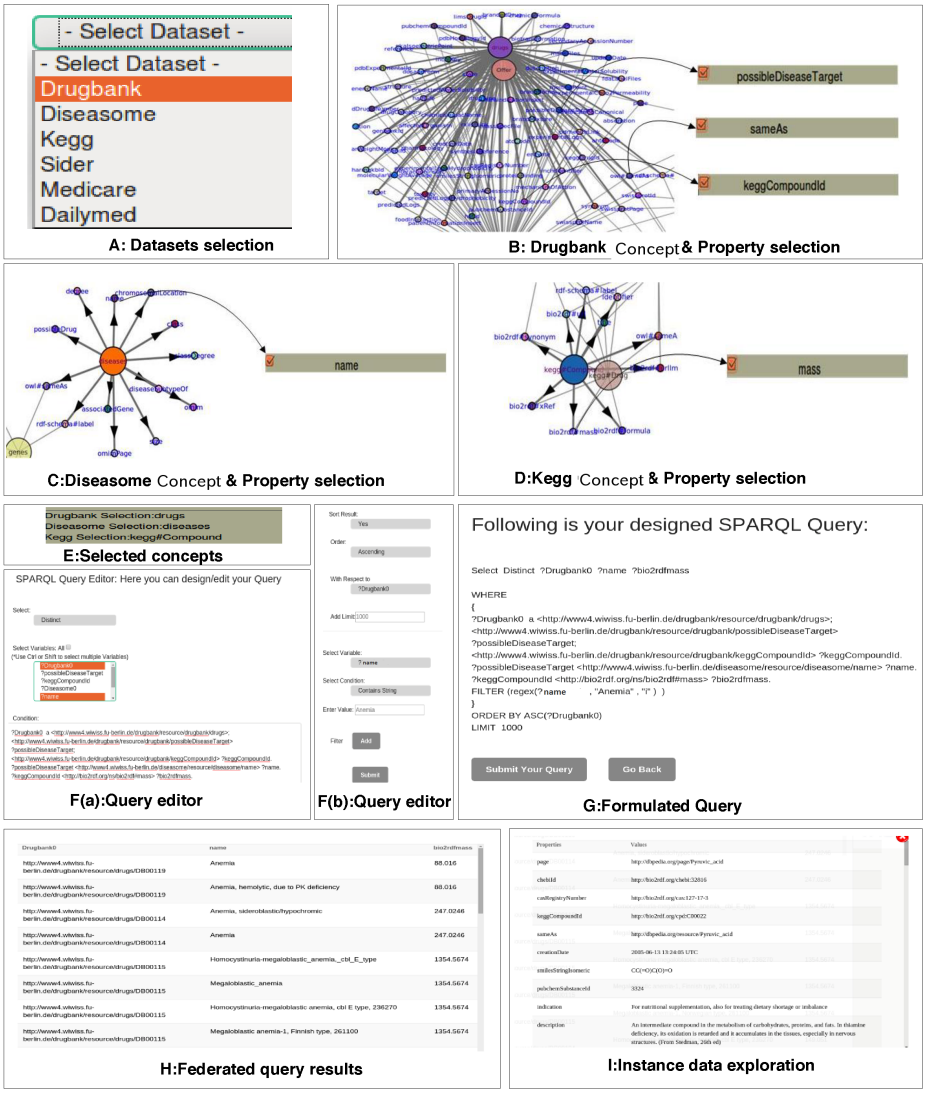

Fig. 3: Federated query formulation using FedViz

Dataset	Triples	Subjects	Predicates	Objects	Classes
DrugBank	517023	19693	119	276142	8
Kegg	1090830	34260	21	939258	4
Dailymed	162972	10015	28	67782	6
Diseasome	72445	8152	19	27704	4
Sider	101542	2674	11	29410	4
Medicare	44500	6825	6	23308	3
Total	1989312	81619	204	1363604	29

Table 1: Dataset Statistics

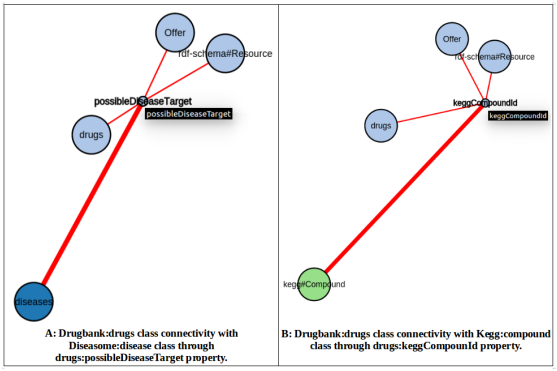

Fig. 4: Datasets Class visualisation view assign each dataset with unique colour. Light Blue: Drugbank, Dark Blue: Diseasome and Light Green: Kegg. Connectivity between Drugbank:drugs with Diseasome:disease through drugs:possibleDiseaseTarget property (Fig 4-A). Connectivity between Drugbank:drugs with Kegg:compounds through drugs:keggCompoundId property (Fig 4-B).

1. The first step is to identify how Drugbank, Diseasome and Kegg datasets are connected to each other? This connectivity (i.e., via classes `drugbank:drug`, `diseasome:disease` and `kegg:compound` can be found by using the Class visualisation view of FedViz that shows all classes of datasets along with there connectivity (ref., Fig. 4).
2. User selects Drugbank from the Datasets Selection box (window A).
3. The visualisation for Drugbank dataset can be seen in window B where he selects `drugbank:drug` class and its properties(i.e., `drugs:possibleDiseaseTarget` and `drugs:keggCompoundId`).
4. Step 2 and 3 are now followed for Diseasome dataset, i.e., select `diseasome:disease` class and it's `name` property (window C) and for Kegg dataset, i.e., select `kegg:compound` class and it's `mass` property (window D).
5. Selected Concepts are shown in status bar (window E).
6. Next, FedViz SPARQL Query Editor allows user to add constraints to the formulated federated query such as select projection variables, apply SPARQL `LIMIT`,

`FILTER`(in this scenario disease name Anemia), `ORDERY BY` clauses, and can further edit the query according to his choice (window Fa, Fb).
7. The final query can be seen on submission (window G).
8. Query is executed over FedX and the retrieved results are displayed by FedViz (Result window H).
9. Finally, by selecting any URI from the retrieved result, FedViz can provide detailed information regarding that instance (Data Exploration window I).

4.4 Query Execution

On dispatching from FedViz, SPARQL query is received and handled by an intermediate layer (IL) built on top of FedX [22]. The IL acts as an adopter, which allows the FedX to communicate with outer world (i.e, Web). FedX requires the set of endpoints URLs as input to query execution engine. The FedViz request incorporates the set of endpoints required by the query. The IL forwards the endpoints to FedX query engine by selecting endpoints from request. FedX executes a SPARQL *ASK* requests on set of endpoints. Furthermore, FedX optimise the query by splitting it into sub-queries. The selected endpoints are requested to run these sub-queries to generate the results. Finally, all the retrieved results from various sub-queries are integrated and displayed through FedViz interface.

5 Evaluation

The goal of our evaluation is to quantify the usability and usefulness of FedViz graphical interface. We evaluate the usability of the interface by using the standard *System Usability Scale* (SUS) [14] as well as a customised questionnaire designed for the users of our system. In the following, we explain the survey outcomes.

5.1 System Usability Scale Survey

In this section, we explain the SUS questionnaire[12] results. This survey is more general and applicable to any system to measure the usability. The SUS is a simple, low-cost, reliable 10 item scale that can be used for global assessments of systems usability[14,17]. As of 10th July 2015, 15 users[13] including researchers and engineers in Semantic Web were participated in survey. According to SUS, we achieved a mean usability score of **74.16**% indicating a high level of usability according to the SUS score. The average scores (out of 5) for each survey question along with standard deviation is shown in Figure 5.

The responses to question 1 (average score to question 1 = 3.8 ± 0.86) suggests that FedViz is adequate for frequent use. The responses to question 3 indicates that FedViz is easy to use (average score 4 ± 0.84) and the responses to question 7 (average score 4.06

[12] SUS survey can found at: http://goo.gl/forms/bhReuNgd6O

[13] Users from AKSW, University of Leipzig and INSIGHT Centre, National University of Ireland, Galway. Summary of the responses can be found at: https://goo.gl/ZOrJx9

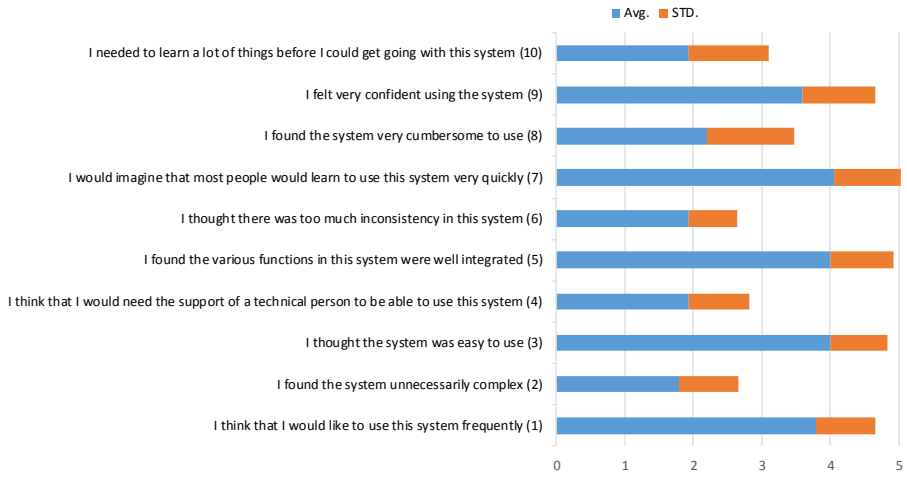

Fig. 5: Result of usability evaluation using SUS questionnaire.

$\pm$ 0.96) suggests that most people would learn to use this system very quickly. However, the slightly higher standard deviation to question 9 (standard deviation = $\pm$ 1.05) and question 10 (standard deviation = $\pm$ 1.16) suggest that we may need a user manual to explain the different functionalists provided by the FedViz interface.

5.2 Custom survey

This survey[14] was particularly designed to measure the usability and usefulness of the different functionalists provided by FedViz. In particular, we asked users to formulate both federated and non-federated SPARQL queries and share their experience through question 10 and question 11. As of 10th July 2015, 10 researchers including Computer Scientist[15] and Bioinformaticians were participated in survey. The average scores (out of 5 with 1 means strongly disagree and 5 means strongly agree) for each survey question along with standard deviation is shown in Figure 6. The average scores to question 10 (i.e., 4.2 $\pm$ 0.91) and question 11 (i.e., 3.9 $\pm$ 0.73) show that most of the user feel confident in formulating simple and federated queries, respectively. The responses to question 2 (average score = 4.4 $\pm$ 0.69) suggests that navigating on different datasets are much easy by using FedViz "Selection Box". A slightly lower scores to question 7 (average score = 3.5 $\pm$ 0.70) suggests that we need to further improve the datasets visualisation component of the FedViz.

As an overall usability evaluation, our SUS and custom surveys outcome suggest that FedViz interface is easy to use, consistent, adequate for frequent use, easy to learn, and the various functions in the system are well integrated.

[14] Custom survey can be found at: http://goo.gl/forms/2DWvK2qYsV
[15] Summary of the responses can be found at: https://goo.gl/tT8TXF

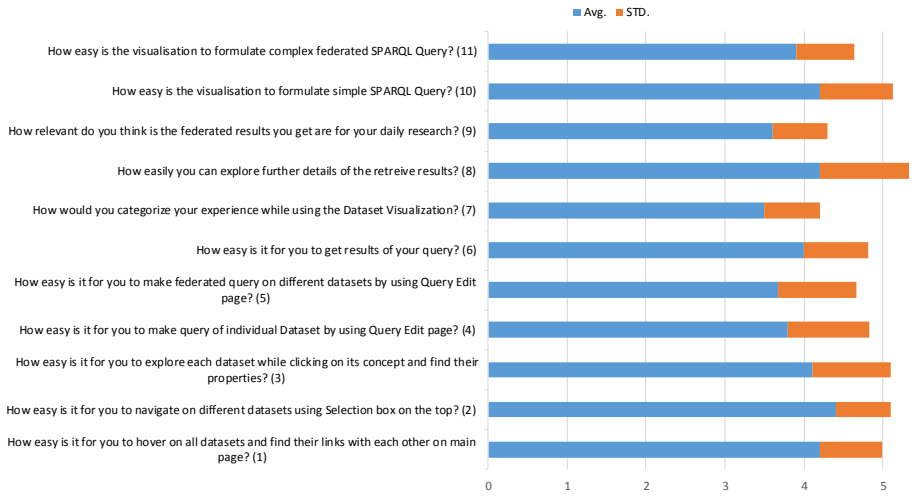

Fig. 6: Result of usefulness evaluation using our custom questionnaire.

6 Conclusion and Future Work

In this paper we introduce FedViz as a online interface for SPARQL query formulation and execution. We evaluate our approach and usability of our system using the standard system usability scale as well as through domain experts. Our preliminary analysis and evaluation revels the overall usability score of 74.16%, concluding FedViz an interface, easy to learn and help users formulating complex SPARQL queries intuitively. As a future work we aim to extend FedViz with Faceted browsing and also provide visualization at entity level e.g, Genes and Molecules where user can see the Gene sequences and 3D structure for Molecules.

7 Acknowledgement

The work presented in this paper has been partly funded by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2).

References

1. J. Almeida, H. Deus, and W. Maass. Development of integrative bioinformatics applications using cloud computing resources and knowledge organization systems (kos). *Nature proceedings*, 2011.
2. F. Belleau, M.-A. Nolin, N. Tourigny, P. Rigault, and J. Morissette. Bio2rdf: towards a mashup to build bioinformatics knowledge systems. *Journal of biomedical informatics*, 41(5):706–716, 2008.
3. J. Borsje and H. Embregts. Graphical query composition and natural language processing in an rdf visualization interface. *Erasmus School of Economics and Business Economics, Vol. Bachelor. Erasmus University, Rotterdam*, 2006.

4. M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2301–2309, 2011.
5. B. Chen, D. J. Wild, Q. Zhu, Y. Ding, X. Dong, M. Sankaranarayanan, H. Wang, and Y. Sun. Chem2bio2rdf: A linked open data portal for chemical biology. *arXiv preprint arXiv:1012.4759*, 2010.
6. H. Chen, T. Yu, and J. Y. Chen. Semantic web meets integrative biology: a survey. *Briefings in bioinformatics*, 14(1):109–125, 2013.
7. F. Haag, S. Lohmann, S. Siek, and T. Ertl. Visual querying of linked data with QueryVOWL. In *Joint Proceedings of SumPre 2015 and HSWI 2014-15*. CEUR-WS, to appear.
8. A. Hasnain, R. Fox, S. Decker, and H. F. Deus. Cataloguing and linking life sciences LOD Cloud. In *EKAW*, 2012.
9. A. Hasnain, M. R. Kamdar, P. Hasapis, D. Zeginis, C. N. Warren Jr, et al. Linked Biomedical Dataspace: Lessons Learned integrating Data for Drug Discovery. In *International Semantic Web Conference (In-Use Track), October 2014*, 2014.
10. A. Hasnain, S. S. E. Zainab, M. R. Kamdar, Q. Mehmood, C. Warren Jr, et al. A roadmap for navigating the life scinces linked open data cloud. In *International Semantic Technology (JIST2014) conference*, 2014.
11. F. Hogenboom, V. Milea, F. Frasincar, and U. Kaymak. Rdf-gl: a sparql-based graphical query language for rdf. In *Emergent Web Intelligence: Advanced Information Retrieval*, pages 87–116. Springer, 2010.
12. D. H. Jonassen, K. Beissner, and M. Yacci. *Structural knowledge: Techniques for representing, conveying, and acquiring structural knowledge*. Psychology Press, 1993.
13. M. R. Kamdar, D. Zeginis, A. Hasnain, S. Decker, and H. F. Deus. Reveald: A user-driven domain-specific interactive search platform for biomedical research. *Journal of biomedical informatics*, 47:112–130, 2014.
14. J. R. Lewis and J. Sauro. The factor structure of the system usability scale. In *HCD*. 2009.
15. A. Russell and P. Smart. Nitelight: A graphical editor for sparql queries. 2008.
16. A. Ruttenberg, T. Clark, W. Bug, M. Samwald, O. Bodenreider, H. Chen, D. Doherty, K. Forsberg, Y. Gao, V. Kashyap, et al. Advancing translational research with the semantic web. *BMC bioinformatics*, 8(Suppl 3):S2, 2007.
17. M. Saleem, M. R. Kamdar, A. Iqbal, S. Sampath, H. F. Deus, and A.-C. N. Ngomo. Big linked cancer data: Integrating linked tcga and pubmed. *Web Semantics: Science, Services and Agents on the World Wide Web*, 27:34–41, 2014.
18. M. Saleem, Y. Khan, A. Hasnain, I. Ermilov, and A.-C. N. Ngomo. A fine-grained evaluation of sparql endpoint federation systems. *Semantic Web Journal*, 2014.
19. M. Saleem, S. S. Padmanabhuni, A.-C. N. Ngomo, A. Iqbal, J. S. Almeida, S. Decker, and H. F. Deus. Topfed: Tcga tailored federated query processing and linking to lod. *Journal of Biomedical Semantics*, 2014.
20. M. Samwald, A. Jentzsch, C. Bouton, C. S. Kallesøe, E. Willighagen, J. Hajagos, M. S. Marshall, E. Prud'hommeaux, O. Hassanzadeh, E. Pichler, et al. Linked open drug data for pharmaceutical research and development. *Journal of cheminformatics*, 3(1):19, 2011.
21. M. Schmidt, O. Görlitz, P. Haase, G. Ladwig, A. Schwarte, and T. Tran. Fedbench: A benchmark suite for federated semantic data query processing. In *The Semantic Web–ISWC 2011*, pages 585–600. Springer, 2011.
22. A. Schwarte, P. Haase, K. Hose, R. Schenkel, and M. Schmidt. Fedx: Optimization techniques for federated query processing on linked data. In *The Semantic Web, ISWC 2011*, volume 7031 of *Lecture Notes in Computer Science*, pages 601–616. 2011.
23. R. Tamassia. *Handbook of graph drawing and visualization*. CRC press, 2013.
24. J. D. Wallace and J. J. Mintzes. The concept map as a research tool: Exploring conceptual change in biology. *Journal of research in science teaching*, 27(10):1033–1052, 1990.