INODE 1st Evalution Report

Document Due Date: 30/04/2021
Document Submission Date: 30/04/2021

Work Packages 2, 3, 4, 5, 6, 7, 8

Type: Report
Document Dissemination Level: Public

INODE
Intelligent Open Data Exploration
is funded by the Horizon 2020 Framework Programme of the EU for Research and Innovation.
Grant Agreement number: 863410— INODE — H2020-EU.1.4.1.3.

(This page has been intentionally left blank

Executive Summary

This deliverable provides the first evaluation report on the INODE system architecture (WP2), and on the design and implementation of all 6 services covered by WPs 3, 4, 5, 6, 7 and 8. The deliverables of all work packages are integrated into one report.

For convenience, an overview of the INODE system architecture with the major services is given in Figure 1. The main interface for users to interact with the INODE-SQL 2.0 part of the system in the present release is the OpenDataDialog 2.0 web application. The details for all components and services are presented in Deliverable 3.2. In the current deliverable, we discuss the results of the evaluation of all components as well as an end-to-end exploration pipeline evaluation.

The services shown in green refer to "OpenDataDialog", the services in orange to "OpenDataLinking" and the services in blue are "Backend Services". INODE-SQL 2.0 is the user-facing service that provides access over SQL data sources, and INODE-SPARQL 1.0 allows access over RDF knowledge graphs.

As much as possible, our contributions are illustrated based on our use case datasets . We also used other datasets that best showcase the features of our solutions. Nevertheless, illustrating all the features of INODE on our use case datasets is ongoing work.

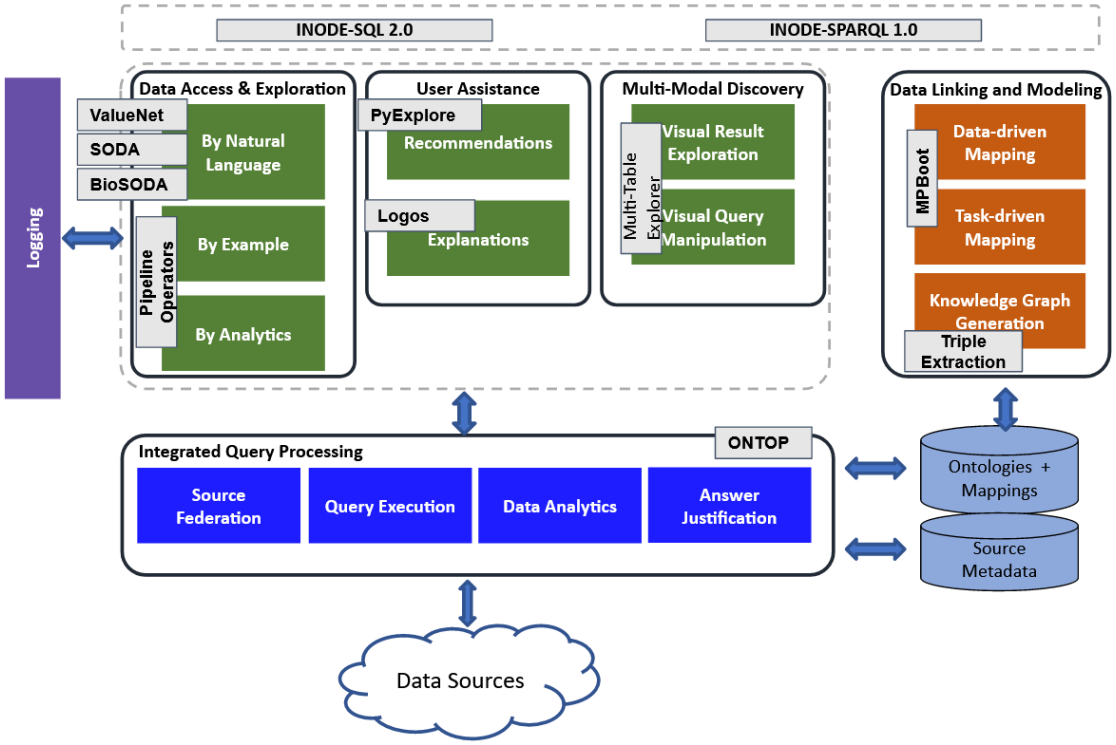


Figure 1: *The INODE system architecture services.*

Project Information

Project Name	Intelligent Open Data Exploration
Project Acronym	INODE
Project Coordinator	Zurich University of Applied Sciences (ZHAW), CH
Project Funded by	European Commission
Under the Programme	H2020-EU.1.4.1.3. - Development, deployment and operation of ICT-based e-infrastructures
Call	H2020-INFRAEOSC-2019-1
Topic	INFRAEOSC-02-2019 - Prototyping new innovative services
Funding Instrument	Research and Innovation action
Grant Agreement No.	863410

Document Information

Authors(s)	Koutrika Georgia, Eleftheraki Stavroula, Glenis Apostolis, Mandamadiotis Antonis, Katsogiannis-Meimarakis George (ATHENA) Amer-Yahia Sihem, Patil Yogendra, Personnaz Aurélien (CNRS) Lücke-Tieke Hendrik, May Thorsten (Fraunhofer) Litke Antonis, Papadakis Nikolaos, Papadopoulos Dimitris (Infili) Fabricius Maximilian, Subramanian Srividya (MPE) Bastian Frederic, Mendes de Farias Tarcisio, Stockinger Heinz (SIB) Massucci Francesco, Multari Francesco, Rull Guillem (SIRIS) Calvanese Diego, Lanti Davide, Mosca Alessandro, Xiao Guohui (UNIBZ) Braschler Martin, Brunner Ursin, Kosten Catherine, Smith Ellery, Stockinger Kurt (ZHAW)

Table of Contents

1 Integrated Query Processing

Integrated Query Processing is the low-level INODE component in charge of providing SPARQL query access to the underlying data sources. This component implements a *Virtual Knowledge Graph*[1] (VKG) approach, which we illustrate in Figure 1.1.

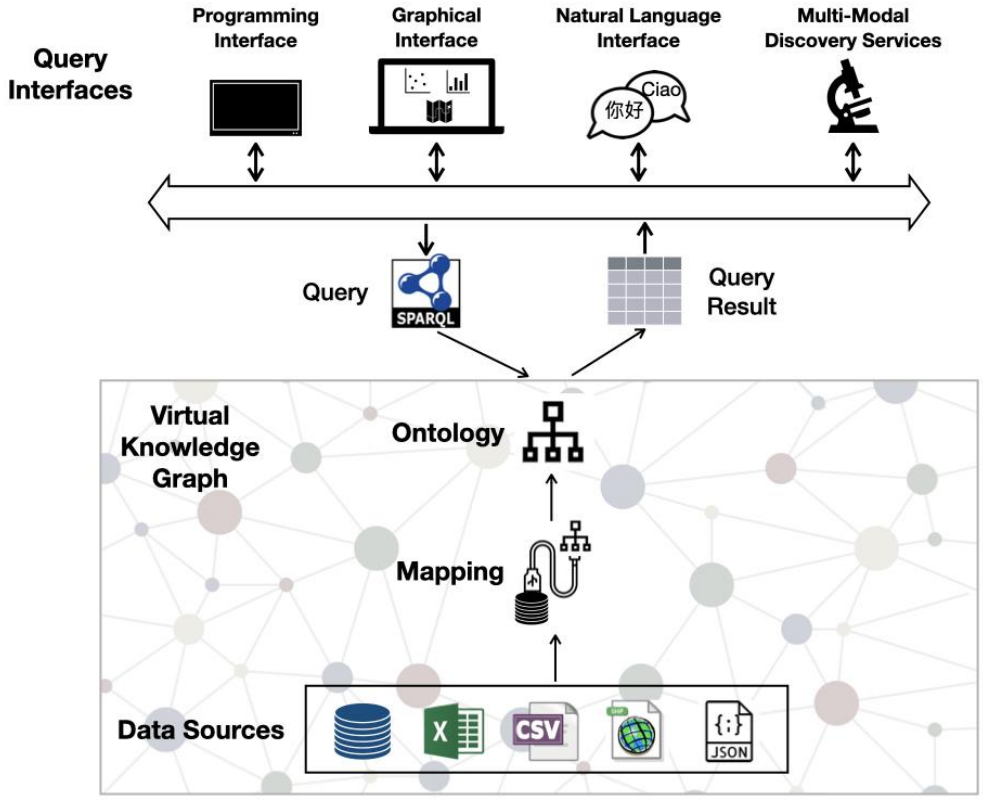


Figure 1.1: *The VKG Framework of the Integrated Query Processing component in INODE.*

In the VKG approach, the data sources (e.g., relational databases) are linked to an *Ontology* providing domain knowledge through a *Mapping*. The goal is to *provide* a (virtual) knowledge graph which constitutes a high-level conceptual view of the data. By querying the VKG, the user can access the information stored in the data sources by means of a more convenient vocabulary, does not need to be aware of storage details, and can obtain richer answers thanks to the domain knowledge.

The INODE Integrated Query Processing component consists of the *VKG system Ontop*[2], which is a state-of-the-art system maintained and developed at the Free University of Bozen-Bolzano. Services are provided to the higher-level components (e.g., the NL-to-SPARQL system Bio-SODA, which will be presented in Section 3) through the W3C standard *SPARQL HTTP protocol*[3].

[1] Guohui Xiao, Linfang Ding, Benjamin Cogrel, & Diego Calvanese (2019). Virtual knowledge graphs: An overview of systems and use cases. *Data Intelligence*, *1*(3), 201-223.
[2] https://ontop-vkg.org/
[3] https://www.w3.org/TR/sparql11-http-rdf-update/

In this section, we provide a thorough evaluation of the Integrated Query Processing component. The section is structured as follows: in Subsections 1.1 and 1.2, we discuss and evaluate the progress with respect to Task 3.1 (*Query Execution over Rich Types of Data Sources*) and Task 3.3 (*Data Analytics*) of the INODE proposal. In Subsection 1.3, we evaluate the SPARQL query answering service over the three use cases of the INODE project.

For Task 3.3, we observe that we have anticipated the work with respect to the foreseen work plan, and therefore have already been able to carry out an evaluation, which was planned for M24. On the other hand, the work on Task 3.2 (*Source Federation*) has been delayed, as explained in Deliverable D1.1. Although we have recently implemented in the Ontop system the support to SQL federation by relying on popular data federation systems (notably Denodo, Dremio, and Teiid), the evaluation of the novel federation functionalities is postponed to M24.

1.1 Query Execution over Rich Types of Data Sources

We have extended Ontop to support geospatial data. In particular, Ontop now supports the querying of GIS[4] data through *GeoSPARQL*[5], a standard query language from the Open Geospatial Consortium. The scientific outcome of this activity are two journal publications in *Geoinformatica*[6] (impact factor 2.161) and in the *International Journal of Geo-Information*[7] (impact factor 2.239). In such research, we have extensively tested the geospatial (and temporal) support by relying on data from the Südtirol Open data portal[8] in order to assess the data quality and apply visual analysis to the considered data sets. The system is currently successfully deployed at the OpenDataHub portal[9]. In such a portal, the user can provide SPARQL queries (also guided by a set of predefined queries that are provided) and, among other things, visualize the answers on a map, as illustrated in Figure 1.2.

[4] GIS = Geographic Information System
[5] https://www.ogc.org/standards/geosparql
[6] Ding, L., Xiao, G., Calvanese, D., & Meng, L. (2019). Consistency assessment for open geodata integration: An ontology-based approach. *Geoinformatica*, 1-26.
[7] Ding, L., Xiao, G., Calvanese, D., & Meng, L. (2020). A Framework Uniting Ontology-Based Geodata Integration and Geovisual Analytics. *ISPRS International Journal of Geo-Information*, *9*(8), 474.
[8] https://civis.bz.it/it/
[9] https://sparql.opendatahub.bz.it/

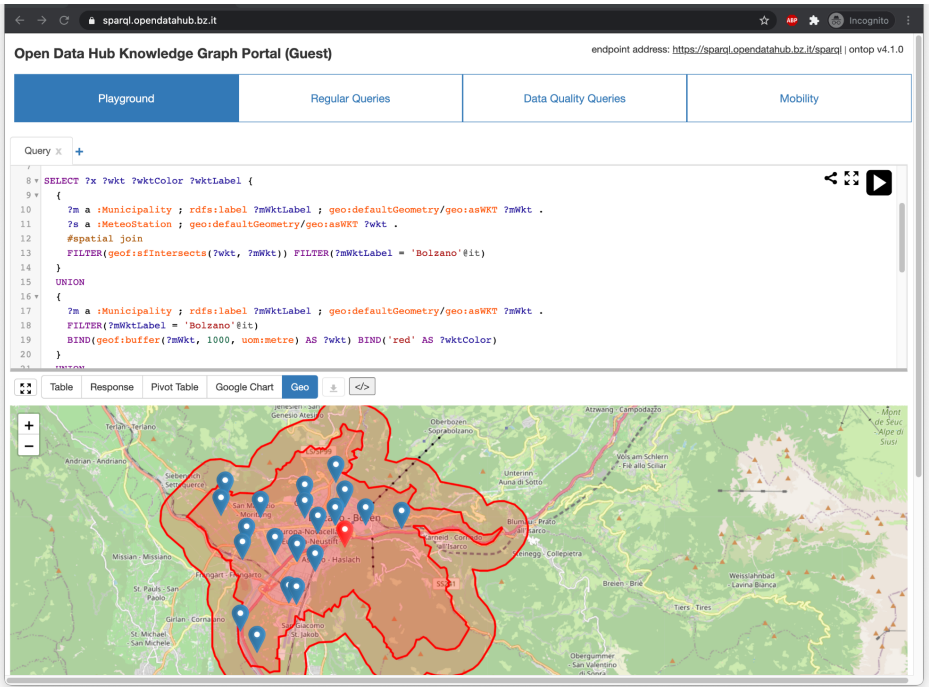


Figure 1.2: *Positions of meteorological stations within 1km from the municipality borders of the city of Bolzano (Italy).*

Since the data extracted for the three use cases in INODE do not contain geospatial information that is compliant with the Open Geospatial Consortium (OGC) standards, we have not relied on them to evaluate the geospatial capabilities of Ontop. However, we foresee applications for geospatial capabilities of Ontop in at least one of the INODE scenarios: in CORDIS, we can integrate geospatial information regarding institutions taken from available RDF repositories (e.g., DBPedia). On the other hand, in astrophysics we do not foresee at the moment an application of geospatial data (which, as the name suggests, describes space *on earth*), nor we do foresee it for the biological case (which deals with the microscopic world).

1.2 Data Analytics

To provide support for analytics tasks, we have implemented in Ontop all SPARQL 1.1 aggregate functions, namely COUNT, SUM, MIN, MAX, AVG, SAMPLE, and GROUP_CONCAT. We point out that providing such support in a VKG setting has required a significant effort. To do so, in fact, we had to rewrite the entire Ontop codebase (for details, see Deliverable D3.2), and shift from an internal representation for queries based on Datalog rules (reflecting the traditional and established theoretical foundations of the VKG approach) to a novel internal algebraic representation that could account also for novel operators. As a result of this effort, to the best of our knowledge, *Ontop is the only open source VKG system providing support for aggregate functions* in full compliance with the SPARQL 1.1 standard.

The scientific outcome of this research effort has been a publication[10] at the *International Semantic Web Conference* (ISWC 2020), the most prestigious venue of the area.

In parallel, we have also carried out an investigation on foundational results over ontology and query languages for queries using the COUNT operator. The scientific outcome of this effort was a publication[11] at the *International Joint Conference on Artificial Intelligence* (IJCAI 2020), one of the most outstanding conference venues in the Artificial Intelligence field.

Table 1.1: *Mean and standard deviation of SPARQL query execution times (in seconds) along with the number of retrieved results for each question, for the test queries that contain aggregate functions over the CORDIS and astrophysics datasets.*

Id	Question	Mean (s)	Std	#Results
CORDIS Q13	Count the ERC projects in the applied life sciences domain	0.134	0.033	1
CORDIS Q19	Total grants received by projects in the area of materials technology	0.232	0.036	1
CORDIS Q30	Find the country with the highest number of projects	1.515	0.075	1
Astro Q5	Count the number of spectra of each spectral classification (galaxy, quasar, star)	3.968	0.242	3

The usefulness of aggregate functions is also recognizable in our use cases. In particular, 4 out of 56 SPARQL test queries (provided in Section 1.3, separately for the three use cases) contain aggregate functions. All 4 queries were executed 10 times, and Table 1.1 depicts the mean and the standard deviation (Std) in seconds for each evaluated query. We can observe that the execution times are in the order of seconds, in line with those of queries without aggregate functions (displayed in Tables 1.2 and 1.3). In other words, support for aggregate functions has been achieved without posing a particular overhead on the system.

1.3 Querying INODE Use Case Datasets with SPARQL

In this section, we evaluate the query answering functionality and performance of Ontop over the three use cases of the INODE project.

After setting the CORDIS, SDSS and OncoMX relational databases to be accessible through a SPARQL endpoint by implementing the VKG approach with Ontop, we evaluated 56 queries in terms of execution time and completeness of the retrieved results. Each query was executed 10 times, and Tables 1.2, 1.3, and 1.4 depict the mean and the standard deviation (Std) in

[10] Guohui Xiao, Davide Lanti, Roman Kontchakov, Sarah Komla-Ebri, Elem Güzel-Kalaycı, Linfang Ding, Julien Corman, Benjamin Cogrel, Diego Calvanese, & Elena Botoeva (2020). The virtual knowledge graph system Ontop. In *Proc. of the 19th Int. Semantic Web Conf. (ISWC)*, pp. 259-277. Springer.

[11] Diego Calvanese, Julien Corman, Davide Lanti, & Simon Razniewski (2020). Counting query answers over a DL-Lite knowledge base. In *Proc. of the 29th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pp. 1658-1666. *IJCAI Org*.

seconds for each evaluated query over the CORDIS, SDSS and OncoMX databases, respectively. Almost all queries were executed in less than one second.

1.3.1 Querying CORDIS with SPARQL

All queries listed in the "Example queries" tab of the INODE testbed[12] that are also shown in Table 1.2 retrieved all expected results, except for 2 failing queries. Q27 and Q28 in Table 1.2 failed because they contain a '`NOT EXISTS`' operator, which is currently not supported by Ontop.

Table 1.2: *The mean and standard deviation of SPARQL query execution times (in seconds) over the CORDIS dataset, along with the number of retrieved results for each question.*

Id	Question	Mean (s)	Std	#Results
Q1	What is the city of opel automobile	0.402	0.053	2
Q2	What is the country code of Latvia	0.098	0.009	1
Q3	Projects funded by the FP7 program	0.785	0.058	25778
Q4	Projects in the area of mathematics	0.175	0.023	239
Q5	Projects in mass spectrometry	0.202	0.050	16
Q6	Show ERC research domains in the diagnostics tools panel	0.164	0.029	426
Q7	What are the participants of the project alfred	0.368	0.023	14
Q8	Starting year of the project theseus	0.156	0.025	4
Q9	Organizations in the awareness project	0.329	0.116	1
Q10	Ending year of projects in the area of climate change	0.141	0.024	62
Q11	Panels of projects in genome editing	0.197	0.030	5
Q12	Projects starting in 2019 with the university of zurich	0.272	0.035	17
Q13	Count the ERC projects in the applied life sciences domain	0.134	0.033	1
Q14	Topics of projects in life sciences	0.664	0.112	4463
Q15	Linguistics projects related to the human mind	0.124	0.037	2
Q16	All projects that started in 2015 in switzerland	0.526	0.079	1066
Q17	ERC projects whose principal investigator is Michael Smith	0.147	0.038	1

[12] http://testbed.inode.igd.fraunhofer.de:18000/

Q18	Grants received by projects in big data	0.417	0.044	1019
Q19	Total grants received by projects in the area of materials technology	0.232	0.036	1
Q20	Projects starting in 2016 whose host is the university of zurich	0.362	0.039	9
Q21	Full name of principal investigators of projects hosted in france	0.507	0.040	586
Q22	Titles of erc projects with coordinators and their geographic location	0.352	0.046	12
Q23	Universities which are coordinators in climate change projects	0.554	0.052	1877
Q24	Countries with no projects	0.906	0.055	62
Q25	Projects with a cost higher than 1 million	1.351	0.120	25744
Q26	Projects started after November 2019	0.141	0.023	451
Q27	Projects including participants from greece and romania	-	-	-
Q28	Projects not including participants from greece nor romania	-	-	-
Q29	Find the project with the highest funding	0.635	0.190	1
Q30	Find the country with the highest number of projects	1.515	0.075	1

1.3.2 Querying SDSS with SPARQL

All queries listed in the "Example queries" tab of the INODE testbed[13] that are also depicted in Table 1.3 retrieved all expected results. Among them, Q7 takes significantly longer, namely 45 seconds. The reason for this is that the query is asking for a "magnitude_g" value, without specifying what kind of magnitude_g is actually requested. According to the fragment of the astrophysics ontology in Figure 1.3, there are actually 7 different kinds of "magnitude_g" applicable here, corresponding to 7 different subproperties of the "magnitude_g" data property: Ontop will produce a translation to the source SQL database which takes into account all of these alternatives, resulting in a query with 6 '**`UNION ALL`**' operators. If we specify one of these subproperties, we can observe that the execution time is significantly reduced (e.g., if in Q7 we ask for "expmagnitude_g" in place of "magnitude_g", then the mean execution time falls to below one second).

[13] http://testbed.inode.igd.fraunhofer.de:18006/

Figure 1.3: *A fragment of the astrophysics ontology: property "magnitude_g" and its seven subproperties.*

Table 1.3 *The mean and standard deviation of SPARQL query execution times (in seconds) over the astrophysics dataset, along with the number of retrieved results for each question.*

Id	Question	Mean (s)	Std	#Results
Q1	Find unique objects in an RA/Dec box	0.198	0.046	318
Q2	Find galaxies with g magnitudes between 18 and 19	0.114	0.027	10
Q3	Rectangular search using straight coordinate constraints	1.789	0.568	12664
Q4	Retrieve both magnitudes (from photometry) and redshifts (from spectroscopy) of quasars	3.193	0.860	100
Q5	Count the number of spectra of each spectral classification (galaxy, quasar, star)	3.968	0.242	3
Q6	Show all spec galaxies with ascension < 130 declination > 5	5.782	0.080	100
Q7	Show all photo galaxies with magnitude_g <= 23 ascension < 130 declination > 5	45.535	2.370	100
Q8	Show all photo asteroids with mode of photo observation 1	12.104	2.861	2
Q9	Show white dwarfs with redshift > 0	0.146	0.037	100
Q10	Show all hot massive blue stars	0.299	0.069	100
Q11	Show all spec stars with plate number 1760	0.537	0.081	15
Q12	Show all spec stars with the subclass WDhotter	0.150	0.036	100
Q13	Show the redshifts of all spectroscoscopies of quasars	0.098	0.040	100
Q14	Show all quasars with ascension > 120 and declination > 5.2	0.770	0.687	100
Q15	Show all star burst galaxies with velocity dispersion > 800	1.235	0.071	100

1.2.3 Querying OncoMX with SPARQL

All queries listed in the "Example queries" tab of the INODE testbed[14] that are also shown in Table 1.4 retrieved all expected results. Among these queries, Q4, Q6, Q7, and Q10 took much longer because Ontop, making use of the mappings, translates these queries into SQL queries that are significantly more complex than if they were written natively in SQL. Experiments have shown that the manually written SQL queries that correspond to Q4, Q6, Q7, and Q10 are executed in less than one second. We are currently investigating how to improve the queries automatically generated by Ontop in order to reduce their execution times.

Table 1.4 *The mean and standard deviation of SPARQL query execution times (in seconds) over the OncoMX dataset, along with the number of retrieved results for each question.*

Id	Question	Mean (s)	Std	#Results
Q1	Cancer single biomarkers and their descriptions	0.132	0.044	931
Q2	Cancer single biomarkers for breast cancer	0.054	0.003	172
Q3	Cancer biomarker panels and their descriptions including indicated cancer type	0.079	0.005	162
Q4	All cancer types in the database	122.006	1.364	43
Q5	All information about species in the database	0.044	0.029	10
Q6	What are the cancer types where the A1BG gene expression is increased (up regulated)?	30.583	0.407	8
Q7	What are the cancer types where the A1BG gene expression is statistically significantly increased (up regulated)?	43.439	0.249	4
Q8	What are the healthy organs where the A1BG is expressed?	0.051	0.028	74
Q9	What are the healthy organs in humans where the A1BG is not expressed?	0.064	0.005	57
Q10	Biomarkers related to breast at the EDRN phase one	23.385	0.177	18
Q11	What are the genomic biomarkers for breast cancer?	0.116	0.016	4

[14] http://testbed.inode.igd.fraunhofer.de:18005/

2 Data Linking and Modelling

In this section, we provide a thorough evaluation of the Data Linking and Modelling component.

2.1 Mapping Construction

In order to enable the VKG approach, we first need to provide an ontology and a *mapping* relating the terms in the ontology to queries over the data sources (recall Figure 1.1). These are offline tasks that need to be carried out before the system is ready to accept queries. Nowadays, the effort of specifying an ontology can be significantly reduced in many domains of interest by importing already existing standard ontologies, or by combining ontology design patterns which can be retrieved from dedicated public catalogs (e.g., the Ontology Design Patterns.org[15] and the Ontology Design Patterns (Odps) Public Catalog[16]). However, the process of specifying mappings that link the elements in the ontology to portions of the data source(s) is usually a labour-intensive activity that needs to be carried out manually. One of the goals of INODE, elaborated in Tasks 4.1 and 4.2 of the proposal, is to develop techniques to support this process. To this aim, we have implemented *MPBoot*, an extension of the *Direct Mapping*[17] W3C standard, according to the *mapping patterns*[18] that we are currently studying within the context of INODE.

A *mapping pattern* puts into correspondence (through mapping assertions) a conceptual portion of an entity relationship (ER) diagram and its typical translation to a logical database schema, with a corresponding encoding into an *OWL 2 QL*[19] ontology. In our research, we have found out that, typically, the majority of the mapping assertions written by VKG engineers and domain experts in real application scenarios can be categorized according to a number of recurrent mapping patterns. In particular for the CORDIS use case, only 9 out of 120 mapping assertions manually written by the ontology engineers do not conform to our categorization into mapping patterns.

The scientific outcome of this activity has been a publication[20] at the *International Conference on Advanced Information Systems Engineering* (CAiSE 2021). In such a work, we present an algorithm, called ADaMAP, to discover a subset of applicable patterns starting from the schema of a relational database. Our algorithm is able to catalog 89 out of 120 mapping assertions, with a precision, recall, and F-measure all equal to 0.8.

MPBoot incorporates some of the ideas of ADaMAP to ease the process of mapping specification. Moreover, it goes beyond the idea of mapping patterns by allowing a number of configurations tailored towards the hybrid (i.e., manually crafted and automatically

[15] http://ontologydesignpatterns.org/wiki/Main_Page
[16] http://www.gong.manchester.ac.uk/odp/html/index.html
[17] https://www.w3.org/TR/rdb-direct-mapping/
[18] Diego Calvanese, Avigdor Gal, Davide Lanti, Marco Montali, Alessandro Mosca, & Roee Shraga (2020). Mapping Patterns for Virtual Knowledge Graphs. *arXiv preprint arXiv:2012.01917*.
[19] The W3C standard for VKGs. Link: https://www.w3.org/TR/owl2-profiles/
[20] Diego Calvanese, Avigdor Gal, Naor Haba, Davide Lanti, Marco Montali, Alessandro Mosca, & Roee Shraga (2021). ADaMaP: Automatic Alignment of Data Sources using Mapping Patterns. In *Proc. of the 33rd Int. Conf. on Advanced Information Systems Engineering (CAiSE 2021)*. Springer. To appear in print.

generated) specification of the mapping assertions. Moreover, it is also able to exploit a *query workload* to produce new mapping assertions (in line with the goals of Task 4.2, "*Task-driven mappings*"). We have successfully adopted MPBoot for the astrophysics scenario: out of 135 mappings, 67 were automatically generated by MPBoot and the remaining ones were manually crafted.

For CORDIS, we have not relied on MPBoot since mappings were already provided by SIRIS Academic before the tool had been finalized. However, as pointed out in our research work published in CAiSE 2021, the majority of mappings in this scenario could have been generated automatically by MPBoot.

For OncoMX, automatically generating mappings is hard. The reason is that, in such a scenario, the terms in the ontology need to be connected to complex view definitions. In other words, it is not possible to "directly" map elements at the source schemas to terms in the desired target ontologies. For such a reason, the mappings for this scenario have been manually crafted by SIB.

Please notice that in these evaluations we have not discussed aspects relative to performance, since the task of mapping construction happens offline and therefore does not affect the overall performance of the (online) query answering services.

2.2 Information Extraction and Knowledge Construction from Text

In this section, we evaluate and compare the performance of our OpenDataLinking Open Information Extraction (OIE) component of INODE-SQL 2.0 focusing on triple extraction from unstructured text, database enrichment via entity linking of the extracted triples with ontology concepts, and runtime performance.

2.2.1 Triple Extraction performance

As mentioned in Deliverables D3.1 and D3.2, we focused on the Cancer Biomarker use case, by extracting triples from PubMed abstracts and mapping these to existing concepts (anatomical entities and genes), aiming at enriching the content of the OncoMX database. A proper evaluation on this specific use case would require a curated list of annotated/gold triples for the ingested PubMed abstracts, which is currently not available. It should also be noted that - even if possible - such an evaluation would only correspond to an extremely narrow topic range, while our intentions are targeted towards an adaptable system with demonstrated generalisability on diverse textual domains.

To overcome the lack of annotated use case-specific datasets for evaluating our approach, we leveraged a number of benchmark datasets that are widely adopted for the evaluation of Open Information Extraction (OIE) systems. We measure the performance of our triple

extractor against two state-of-the-art OIE systems, OpenIE6[21] and IMoJIE[22], on two standard benchmarking annotated datasets, namely CaRB[23] and Re-OIE2016[24].

We report the performance of our triple extraction system in terms of area under the curve (AUC), precision, recall and F1-score, using the CaRB Evaluator[25] on the CaRB test set, and a version of the Re-OIE16 annotations adapted for the CaRB Evaluator. The results are shown in Table 2.1.

Table 2.1 *Evaluation results of our Information Extraction engine on the CaRB test set and Re-OIE16, compared to OpenIE6 and IMoJIE.*

	CaRB				*Re-OIE16*			
	AUC	Precision	Recall	F1	AUC	Precision	Recall	F1
IMoJIE	.333	.647	.456	.535	.483	.653	.584	.617
OpenIE6	.337	.589	.477	.527	.523	.642	.612	.627
OpenDataLinking OIE (our approach)	.390	.600	.488	.538	.541	.668	.648	.658

We observe the most significant improvements in the AUC scores, with an approx. 6% increase over both OpenIE6 and IMoJIE in CaRB, and an increase across all metrics in Re-OIE16. The precision/recall balance achieved by leveraging both rule-based and learning-based extraction approaches combined with post-processing triple refinement techniques is reflected in the improved F1-scores on both datasets. In particular, precision and recall are well-matched on Re-OIE16, despite the system being tuned on the CaRB development set, which demonstrates good generalisability. During the course of the project, we could also consider a more use case-specific evaluation of our information extraction engine based on a manually annotated subset (gold triples) of the utilized corpora.

2.2.2 Database Enrichment and Runtime Evaluation

In order to quantify the effect of our tight integration between triple extraction and entity linking, we also perform a comparison between each system on the Pubmed abstracts

[21] Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Soumen Chakrabarti, et al. (2020). OpenIE6: Iterative Grid Labeling and Coordination Analysis for Open Information Extraction. *arXiv preprint* arXiv:2010.03147 (2020).

[22] Keshav Kolluru, Samarth Aggarwal, Vipul Rathore, Soumen Chakrabarti, et al. (2020). IMoJIE: Iterative Memory-Based Joint Open Information Extraction. *arXiv preprint* arXiv:2005.08178 (2020).

[23] Sangnie Bhardwaj, Samarth Aggarwal, and Mausam Mausam (2019). CaRB: A Crowdsourced Benchmark for Open IE. In *Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int. Joint Conf. on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 6262–6267. https://doi.org/ 10.18653/v1/D19-1651

[24] Junlang Zhan and Hai Zhao (2020). Span model for open information extraction on accurate corpus. In *Proc. of the AAAI Conf. on Artificial Intelligence*, Vol. 34. 9523–9530.

[25] https://github.com/dair-iitd/CaRB

dataset. For this experiment, we run OpenIE6, IMoJIE and OpenDataLinking OIE on a subset of 1,000 PubMed abstracts (3,035 sentences), and perform the entity linking procedure with Uberon and OncoMX concepts, as described in Deliverable D3.1. For OpenIE6 and IMoJIE, we use an n-gram-based search over the Uberon and OncoMX databases, as they do not provide the annotated triples of our system. This procedure simply searches the databases with all possible n-grams from the subject and object, until the longest match is found, rather than deriving the n-grams from the syntactic structure of the triple.

We consider as linked triples only the ones that contain both an Uberon anatomical entity and an OncoMX gene symbol, with one in the subject and the other in the object. Partial matches are not recorded. We also show the average speed of each system, tested on an Intel Core i7-7700HQ 2.80GHz CPU, with 32GB RAM and NVIDIA GeForce GTX 1050 GPU. The results are shown in Table 2.2.

Table 2.2 *Database enrichment and runtime evaluation of our Information Extraction engine on a sample of 1,000 PubMed abstracts, compared to OpenIE6 and IMoJIE.*

	Extracted triples	Linked Triples	Seconds per sentence
IMoJIE	4565	49	14.19
OpenIE6	6675	50	1.46
OpenDataLinking OIE (our approach)	5648	71	7.54

Our results indicate a slower runtime for our system compared to OpenIE6 (which can be attributed to the fact that our implementation consists of several linguistics-based and learning-based extractors), and a faster runtime than IMoJIE. More importantly, we extract more accurate (higher F1 and AUC scores as shown in Table 2.1) and more usable (higher ratio of linked triples as shown in Table 2.2) triples overall.

3 Data Access and Exploration

In this section, we provide a thorough evaluation of the Data Access and Exploration component.

3.1 Querying By-Example and By-Analytics

We present the evaluation of by-example and by-analytics operators. These operators are integrated into pipelines that are trained as reinforcement learning policies that explore the balance between familiarity and curiosity[26]. The by-example operators that we consider in this evaluation mimic the traditional drill-down (using by-facet) and roll up (using by-superset) operators in data exploration[27]. The by-analytics operators that we consider are by-neighbors and by-distribution. The definitions of our by-example operators result in exploring objects that are familiar with the input. The definitions of our by-analytics operators result in exploring objects that are farther from the input (akin to curiosity). *Consequently, our evaluation studies the interplay between expressive data exploration operators (traditional vs all operators) and curiosity-based reinforcement learning.*

Our evaluation shows that the trained pipelines tend to alternate between curiosity- and familiarity-based policies that prioritize one over the other and as the amount of total reward evolves, priorities shift. This illustrates the importance of optimizing for familiarity and curiosity in tandem which justifies the need for all the operators we defined.

This work will appear in the Fourth International Workshop on Exploiting Artificial Intelligence Techniques for Data Management (aiDM), co-located with ACM SIGMOD 2021.[28]

The prototype is available online.[29]

Dataset. We used 2.6 million galaxies in SDSS with clean photometry and spectral information. Each galaxy is described with 7 attributes commonly used in astronomy from two join tables photoobj and specobj. Each column was binned into 10 equi-depth bins. Since our operators are set-based, we formed sets of galaxies where each set contains homogeneous galaxies. To instantiate this model, we used LCM[30] with a support value of 10,

[26] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, Trevor Darrell: Curiosity-driven Exploration by Self-supervised Prediction. ICML 2017: 2778-2787
Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, Koray Kavukcuoglu: Asynchronous Methods for Deep Reinforcement Learning. ICML 2016: 1928-1937

[27] Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya G. Parameswaran, Neoklis Polyzotis: SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics. Proc. VLDB Endow. 8(13): 2182-2193 (2015)
Kyriaki Dimitriadou, Olga Papaemmanouil, Yanlei Diao: AIDE: An Active Learning-Based Approach for Interactive Data Exploration. IEEE Trans. Knowl. Data Eng. 28(11): 2842-2856 (2016)

[28] Aurélien Personnaz, Sihem Amer-Yahia, Laure Berti-Equille, Maximilian Fabricius and Srividya Subramanian: Balancing Familiarity and Curiosity in Data Exploration with Deep Reinforcement Learning. aiDM 2021 (to appear)

[29] http://www.inode-project.eu:18081/test/galaxies.html

[30] Takeaki Uno, Masashi Kiyomi, Hiroki Arimura: LCM ver. 2: Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets. FIMI 2004

and generated 348,857 sets whose size ranges from 10 to 261,793 galaxies. The data was used as an in-memory Pandas dataframe to train the agents in reasonable time. The pipeline operators and item set representations were implemented in Python.

Evaluation task. The goal of the task is to visit as many galaxy types as possible. To encourage agents to visit a maximum number of galaxy types during the exploration, we designed the target set used for training to be "scattered" in the data space. The set was composed by picking 100 samples from 170 types of galaxies defined in the Galaxy Zoo classification[31] (a citizen science project with over 16 million morphological classifications of 304,122 galaxies drawn from the Sloan Digital Sky Survey), resulting in a target set containing 17,000 galaxies (0.65% of the total data).

Training. The agents were trained to generate pipelines under different conditions of curiosity and familiarity. Each agent was trained by starting from a set of galaxies and returned one pipeline. Agents were trained on multiple servers and desktop computers. Training took 100 hours for about 1,700 episodes with 250 steps (operator selection and execution) per episode. Each agent used 6 workers in parallel; the update interval (i.e., number of steps before a policy update) was set to 20 steps and we concatenated five successive states for the LSTM[32] layers of the networks. Training data was stored using wandb[33]. Training resulted in 5 different pipelines:

- FAMO for familiarity-only (this is expected to favor by-example operators and mimics exiting data exploration work)
- CURO for curiosity-only (this is expected to favor by-analytics operators)
- 50FAM-50CUR for 50% familiarity and 50% curiosity
- 75FAM-25CUR for 75% familiarity and 25% curiosity
- 25FAM-75CUR for 25% familiarity and 75% curiosity.

Offline Training Results. Figure 3.1 shows the evolution of familiarity and curiosity rewards, respectively, with traditional and all-operator. The legend for the figure is given above, e.g. FAMO refers to the blue line. Surprisingly, both FAMO and CURO policies produce a mix of familiarity and curiosity rewards. FAMO produces some curiosity reward at the beginning of the training, as every state it goes through is unknown. This reward quickly decreases as FAMO focuses on refining its data familiarity strategy and builds its tour around the data. On the other hand, as CURO explores the data, it finds target objects fortuitously, generating a moderate amount of familiarity reward. Our second observation is that, although FAMO has the best results for familiarity, in every other case, both CURO and FAMO under-perform when compared to other pipelines. For both reward types, and both operator modes, the highest rewards are reached by agents with mixed rewards.

[31] https://www.zooniverse.org/projects/zookeeper/galaxy-zoo/
[32] https://en.wikipedia.org/wiki/Long_short-term_memory
[33] https://github.com/wandb/client

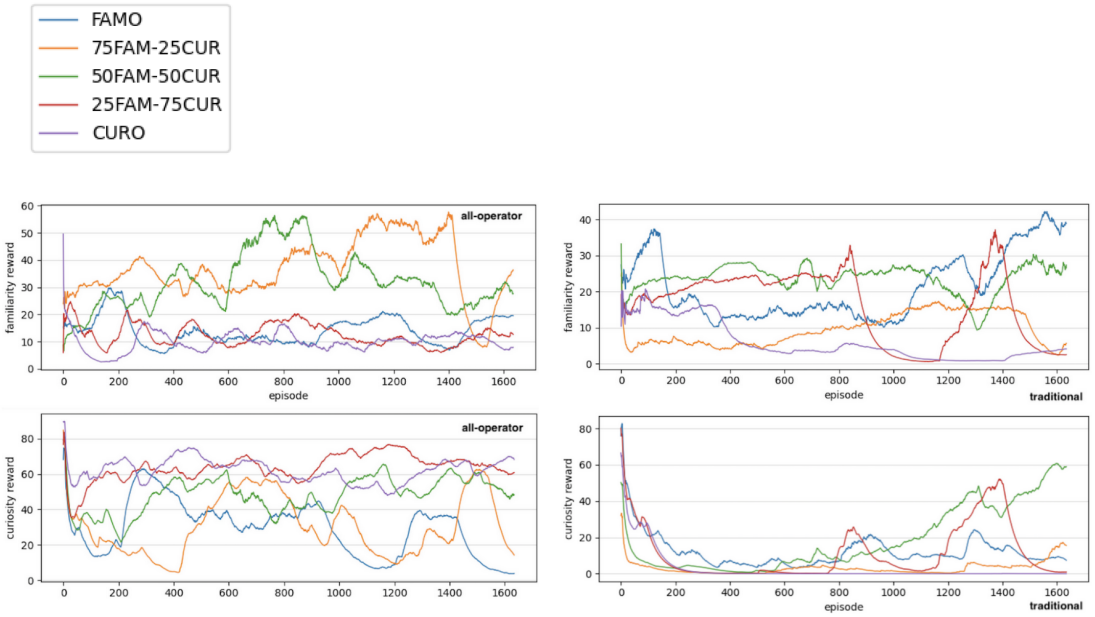


Figure 3.1: *Evolution of familiarity and curiosity rewards respectively with all-operator and traditional training (offline phase)*

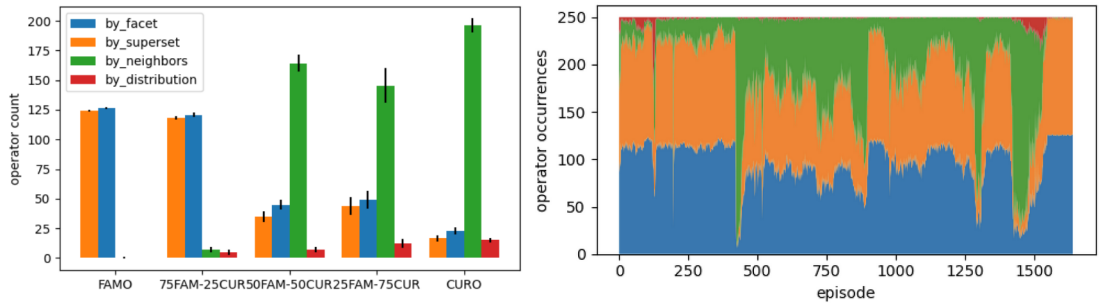


Figure 3.2: *Frequency of use of operators in exploration pipelines (online phase)*

That is particularly noticeable with by-example operators (referred to as traditional), where CURO rapidly runs out of reward and lacks motivation to develop a working policy, while 50FAM-50CUR and 75FAM-25CUR end up with relatively successful curiosity-driven strategies. Similarly, for familiarity with both by-example and by-analytics, we observe that 75FAM-25CUR and 50FAM-50CUR largely outperform FAMO. It is quite the opposite for familiarity-driven policies that do not benefit from more expressive operators. Indeed, we

can see that FAMO reaches much higher performances with traditional operators than with all-operator, where it is outperformed by agents with a mixed reward.

Online Exploration Results. We observe in Figure 3.2 that by-facet and by-superset operators are predominantly selected in familiarity-driven policies such as FAMO and 75FAM-25CUR, whereas by-neighbors and by-distribution operators are preferred in pipelines generated by curiosity-driven policies such as CURO and 25FAM-75CUR. This confirms that, for different weights, the agents will adopt the operators that best support their strategy. This further motivates studying the interplay between data exploration operators and curiosity-driven reinforcement learning in data exploration.

Finally, Figure 3.3 shows the evolution of familiarity and rewards respectively. The results are largely compatible with the offline phase. We can see that in both all-operator and traditional, mixed reward agents clearly outperform FAMO, and that CURO is the worst performer on cumulated familiarity. The curiosity evolution figures corroborate that curiosity-based reward is widely produced by every variant with all-operator, while only 50FAM-50CUR manages to produce curiosity reward with traditional.

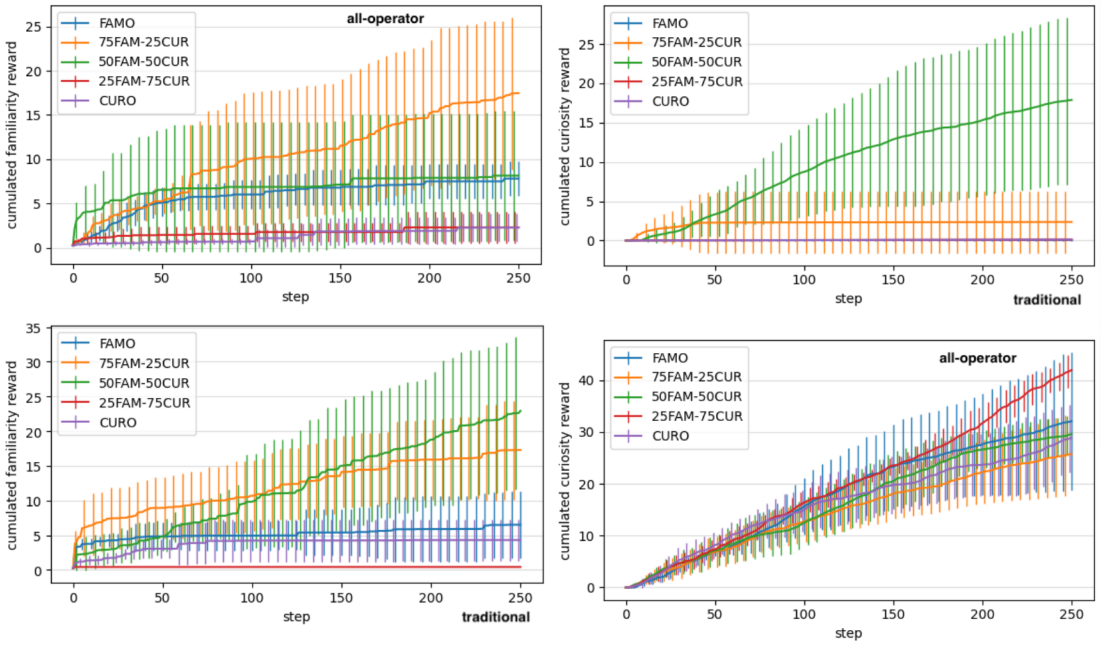


Figure 3.3: *Evolution of familiarity and curiosity rewards respectively with all-operator and traditional pipeline deployments (online phase)*

3.2 Querying By Natural Language

3.2.1 NL-to-SQL: Translating Natural Language Questions to SQL with ValueNet

In this section, we evaluate the performance of ValueNet for translating natural language questions to SQL using the CORDIS dataset. We first present a distilled version of our base model results from the original ValueNet paper[34] that was accepted at International Conference on Data Engineering (ICDE), which is considered to be among the top three most prestigious conferences on database research. Then we show a comparison of the system performance on our INODE use case dataset (CORDIS), with both zero-shot and few-shot learning results.

ValueNet Base Model

In the original ValueNet experiments, we used the Spider[35] dataset which contains 10,181 natural language questions and their SQL equivalents. The queries are spread over 200 publicly available databases from 138 domains. Each database has multiple tables, with an average of 5.1 tables per database. The performance of Valuenet on this dataset provides us with a baseline with which we can compare how well our system performs *transfer learning* on queries from our INODE use case dataset, CORDIS.

We evaluated the ValueNet base model on the Spider dataset using the *Execution Accuracy* metric, which requires executing the synthesized query against a database and comparing if the result is the same as when executing the gold query.

The ValueNet base model achieves an **accuracy of up to 62%** after 100 epochs of training. At the time of writing the original ValueNet paper, ValueNet was among the highest performing approaches in the Spider Challenge[36]. More experimental results along with a detailed error analysis can be found in the long version of the paper[37]. To reproduce our experiments we release all code including hyperparameters on Github[38].

In the following section, we evaluate how well the Valuenet base model performs on a complex real-world dataset (CORDIS) in zero-shot and few-shot settings.

Querying CORDIS

In this section we evaluate ValueNet's ability to transfer its knowledge to a new, unseen database like CORDIS. To do so, we use two different settings:

- A **zero shot** setting, where the ValueNet model has been trained on 146 Spider databases and is then evaluated on 53 questions from CORDIS. In this setting, ValueNet has never seen the CORDIS data or schema before.

---

[34] Brunner, U., & Stockinger, K. (2021). ValueNet: a natural language-to-SQL system that learns from database information. In *International Conference on Data Engineering (ICDE), Chania, Greece, 19-22 April 2021*. IEEE.

[35] Yu, T., Zhang, R., Yang, K., Yasunaga, M., Wang, D., Li, Z., ... & Radev, D. (2018). Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*.

[36] https://yale-lily.github.io//spider

[37] Brunner, U., & Stockinger, K. (2020). ValueNet: a natural language-to-SQL system that learns from database information. arXiv preprintarXiv:2006.00888, 2020

[38] https://github.com/brunnurs/valuenet

- A **few shot** setting, where ValueNet is trained on the Spider data plus 210 CORDIS specific training samples.

The goal is to evaluate how well ValueNet performs on a new, unseen database and if this performance can be improved by adding a small amount of target-specific training samples.

Evaluation Data

The evaluation data consists of 53 handwritten NL question/SQL-query pairs, based on the CORDIS schema & data. The 53 queries cover a large spectrum of SQL features (e.g. JOINS, nested queries, aggregations) and are on average more difficult than the queries from the Spider evaluation data. See evaluation data [39] for more details.

Zero Shot Setting

After training ValueNet on the Spider data (146 databases, 8,659 training samples) we evaluate the trained model on the CORDIS evaluation data. We report an **accuracy of 43%** (23 of 53) correct queries. Please note that in this setting, ValueNet has seen neither the CORDIS schema nor the CORDIS data prior to the evaluation.

We analyse the failing 57% queries and conclude that most of them fail either due to the selection of incorrect columns/tables or to the hard questions (keep in mind that a large part of the evaluation questions can be classified as hard/extra hard according to the Spider difficulty metric). We assume that a few-shot setting, where ValueNet has the possibility to peek into the CORDIS schema/data, should solve the errors (incorrect columns/tables) in the first category.

Few Shot Setting

In this second approach, we first create 210 CORDIS specific training samples. To do so, we use a new approach where random queries are generated based on the database schema and data (see Figure 3.4*)*. We then ask human labelers to describe the visual representation of the query with a question in natural language. For more details see the paper "A Methodology for Creating Question Answering Corpora Using Inverse Data Annotation"[40].

[39] https://github.com/brunnurs/valuenet/blob/d5ddcb168a2c8086de5b7c781f6dc686ad183d09/data/evaluation_pairs_cordis.json

[40] Deriu, J. M., Mlynchyk, K., Schläpfer, P., Rodrigo, A., von Grünigen, D., Kaiser, N., ... & Cieliebak, M. (2020). A methodology for creating question answering corpora using inverse data annotation. In *ACL 2020, Virtual, 5-10 July 2020* (pp. 897-911). Association for Computational Linguistics.

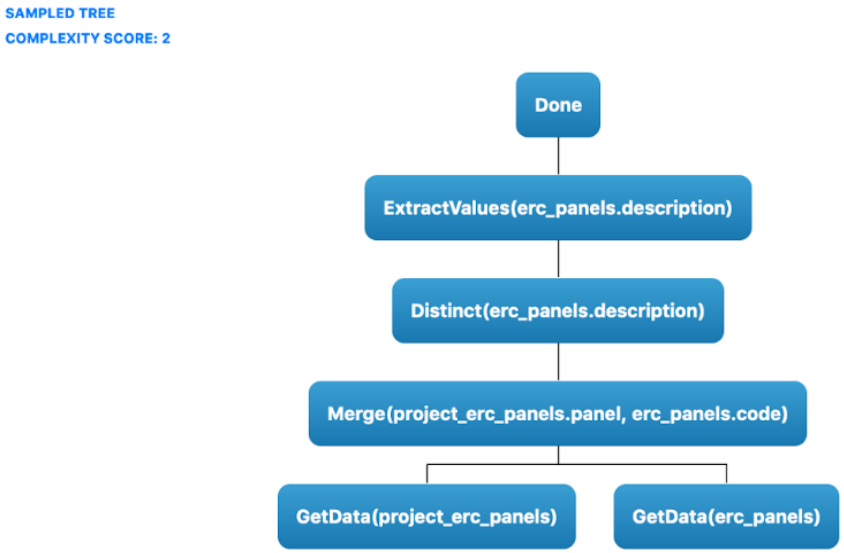


***Figure 3.4**: A generated visual representation of a query in the form of an operator tree. In a second step, the human labeler describes it with a natural language question, e.g. "Show me different descriptions of panels which are used in projects." The question/query pairs are then used as training data.*

After gathering 210 CORDIS-specific training samples with the help of several INODE team members, we train ValueNet on the Spider data (8,659 query/question pairs) as well as the additional 210 CORDIS-specific samples. We do not add any additional weight to the CORDIS samples, so they only account for 2.5% of the total queries.

We evaluate the trained model on the evaluation data and report an **accuracy of 52.8%** (28 of 53) correct queries.

We analyse the failing 47% of queries and see the following issues:

- **Overfitting:** We see several patterns emerge in the sampled training data. For example, the column *"member_short_name"* appears 16 times, whereas the column *"member_name"* does not appear at all. This overfit on specific columns, tables and even JOIN patterns leads to multiple failing queries. Using more and better distributed training data will help mitigate this problem.
- **Hard queries:** several of the failing queries are classified as hard or extra hard. Further improvements in ValueNet will help to handle hard and extra hard queries more reliably.

Given a performance improvement of almost 10% for few shot learning with respect to zero shot learning, we demonstrate that fine tuning on a specific database is a viable approach for getting maximal performance out of a generic model for our CORDIS use case.

3.2.2 NL-to-SPARQL: Translating Natural Language Questions to SPARQL with Bio-SODA

In this section, we evaluate the performance of Bio-SODA, a Knowledge Graph question answering system (KGQA) that translates natural language questions to SPARQL. For a

detailed description of this system, please refer to our previous deliverable, D3.2. The following sections detail Bio-SODA's performance on the test datasets developed from the INODE project use case datasets, CORDIS and SDSS.

Querying CORDIS

The CORDIS NL/SPARQL test data set used to evaluate Bio-SODA consists of 30 NL question/SPARQL query pairs[41] derived from the CORDIS RDF dataset as seen in Section 1.4.1. Note that these queries are different from the ones used for the SQL-based evaluation with ValueNet to cover specific aspects of the SPARQL query language. These queries have on average 2.3 triple patterns per query, which is similar to other KGQA datasets. This data set also has several questions that require 4-6 triple patterns in a query, which is not typical of the queries in the popular KGQA benchmark datasets, but very typical of real world SPARQL queries. Additional complexity of this dataset comes from queries with filters, literals and the ambiguity of the NL questions themselves.

We measure the accuracy of Bio-SODA by comparing the result set of the top ranked SPARQL query against that of the ground truth query. Our evaluation shows that Bio-SODA achieves an **accuracy of 66.7%** (20 of 30) correct queries on the CORDIS NL/SPARQL test set.

The remaining 33.3% (10 of 30) questions that fail contain features that are not currently supported by Bio-SODA such as superlatives, aggregations, comparatives, conjunctions. These types of questions will be supported in a future INODE software release by another NL-to-SPARQL system, ValueNet4SPARQL.

Querying SDSS

Because the majority of the dataset is numeric data, certain operators for comparatives (>, <, =>, =<) were introduced to Bio-SODA for SDSS. These enable additional expressivity for queries with numeric data, which are, because of the nature of this dataset, significantly more common.

For our evaluation of Bio-SODA against the SDSS dataset, we use the same 15 NL/SPARQL query pairs[42] as previously shown in Section 1.4.2. The evaluation dataset for SDSS consists of queries with comparatives, filters, aggregations, literals and an average of 3 triple patterns per query.

We use the same accuracy measure as the evaluation above, to determine the performance of Bio-SODA on the SDSS data. Our evaluation shows that Bio-SODA achieves an **accuracy of 60%** (9 of 15) questions on the SDSS test set. The remaining 40% (6 of 15) questions that fail feature questions with concepts that are not present in the dataset, such as in the question "Rectangular search using straight coordinate constraints". Neither the terms "rectangular search" nor "straight coordinate constraints" are classes or properties of the dataset. Other failed queries include aggregations, which are not supported by Bio-SODA. Further development of the SDSS dataset to include more NL concepts that are used in queries would improve the performance of NL-to-SPARQL systems.

[41] http://biosoda.cloudlab.zhaw.ch:8084/soda/?page=demo

[42] http://testbed.inode.igd.fraunhofer.de:18006/

3.2.3 Hybrid NL-to-SQL: Evaluation of NL-to-SQL approaches

NL-to-SQL systems allow users to explore relational databases by posing free-form queries, alleviating the need for using structured query languages, such as SQL. Existing systems use different approaches and have different query capabilities. For example, some systems support keyword-based queries, other ones only consider simple cases of queries over a single table, and so on. To build NL-to-SQL systems by combining the best of current approaches (i.e., into hybrid approaches as described in Task 5.3), we need to understand the capabilities of these systems in depth.

Existing efforts can be roughly grouped into three categories: (a) Database (DB) approaches, such as SODA, Precis[43] and Discover[44], that leverage the database schema and data to map a query to SQL, (b) Parsing-based approaches, such as NaLIR[45], that parse the input question and use the generated information about the structure of the question to understand its grammatical structure, and (c) Neural machine translation (NMT) approaches, such as ValueNet and Hydranet, which map the text-to-SQL problem to a language translation problem.

Each approach has certain advantages and disadvantages. DB approaches can effectively handle a variety of query types, containing joins, aggregates and nesting, without employing complex neural networks that are time-consuming to train and more cumbersome to deploy. Furthermore, they always produce queries that can be executed over the underlying data. Parsing-based approaches generate a parse tree that contains information about single tokens and their relationships. The parse tree can be easily mapped to query generation rules. NMT approaches have the potential of generalization, i.e., translating more types of NL queries. However, they do not consider the actual data, hence the resulting SQL, even if it is syntactically correct, it may not be executable over the data.

In contrast to evaluation efforts such as WikiSQL and Spider that measure effectiveness based on the number of queries translated to SQL, *we focus on query expressivity,* i.e., the types of queries each system can handle. For this purpose, we designed a NL-to-SQL benchmark that covers several classes of queries. These classes aim at capturing different cases of text queries (e.g., containing typos or synonyms) as well as cases of SQL queries (such as queries with joins, nesting, and so forth). Our effort to evaluate query expressivity complements efforts such as Spider that focus on scale and do not provide such refined query categorization. In particular, we built a rich query benchmark consisting of 216 keyword-based and 241 natural language queries, divided into 17 categories and spanning 3 datasets of varying sizes and complexities.

[43] Alkis Simitsis, Georgia Koutrika, and Yannis Ioannidis. 2008. Précis: from unstructured keywords as queries to structured databases as answers. The VLDB Journal 17, 1 (2008), 117–149

[44] Vagelis Hristidis, Luis Gravano, and Yannis Papakonstantinou. 2003. Efficient IR-style Keyword Search over Relational Databases. In VLDB. 850–861.

[45] Fei Li and H. V. Jagadish. 2014. Constructing an Interactive Natural Language Interface for Relational Databases. PVLDB 8, 1 (Sept. 2014), 73–84

	Category
C1	No joins & no metadata
C2	Joins & no metadata
C3	No joins & metadata
C4	Joins & metadata
C5	Aggregates
C6	GroupBy
C7	Numeric constraints
C8	Logical Operations
C9	Nested
C10	Metadata synonyms
C11	Value synonyms
C12	Metadata misspellings
C13	Value misspellings
C14	Metadata stemming
C15	Value stemming
C16	Negation
C17	Inference logic

Figure 3.5: *The query categories of our benchmark.*

We compared the operation of SODA, NaLIR and other milestone systems that use a database or a parsing-based approach, and we performed extensive experiments that evaluate each system wrt effectiveness, efficiency (i.e., execution time, resource consumption, and scalability) and disambiguation flow.

Table 3.1 presents effectiveness results for the different query categories. We observe that different systems are able to answer only a subset of the query categories, while the most difficult categories, e.g., queries with negation or inference, are not handled at all. Our study also showed that different datasets present different text-to-SQL challenges. We tested both with queries from our general query benchmark as well as for queries using our query categorization for CORDIS and SDSS.

For example, CORDIS attributes are mostly textual and they have descriptive names, like member_name and country. Hence, one would expect that this is an “easy” database for a text-to-SQL system. However, we found that: (a) database normalization has led to several joins to connect necessary information, and (b) foreign keys have similar names with the tables that contain the primary key, which may lead to wrong mappings. On the other hand, SDSS consists of numerical data. SODA and NaLIR answer queries with numerical constraints. However, SDDS attribute names are not self-explanatory. Names such as lnlstar_g are hard for automatic disambiguation methods (such as the ones used by NaLIR) and require an ontology-based approach like the one we are developing in INODE.

Table 3.1: *Average effectiveness percentages.*

(a) Effectiveness results for query categories C1-C4.

	C1			C2			C3			C4		
	top1	top3	top5	top1	top3	top5	top1	top3	top5	top1	top3	top5
Discover	87	100	100	30	60	75	-	-	-	-	-	-
DiscoverIR	87	100	100	25	55	75	-	-	-	-	-	-
Spark	47	60	60	25	40	50	-	-	-	-	-	-
ExpressQ	73	80	80	10	60	70	75	88	88	18	50	64
SODA	73	100	100	45	55	55	63	75	81	36	59	64
NaLIR	-	-	-	-	-	-	67	-	-	71	-	-

(b) Effectiveness results for query categories C5-C9.

	C5			C6			C7			C8			C9		
	top1	top3	top5	top1	top3	top5	top1	top3	top5	top1	top3	top5	top1	top3	top5
ExpressQ	46	75	82	71	88	88	-	-	-	-	-	-	40	40	40
SODA	36	54	54	41	65	71	53	53	73	-	-	-	-	-	-
NaLIR	64	-	-	76	-	-	81	-	-	9	-	-	20	-	-

(c) Effectiveness results for query categories C10-C15.

	C10			C12			C14		
	top1	top3	top5	top1	top3	top5	top1	top3	top5
ExpressQ	-	-	-	-	-	-	-	-	-
SODA	-	-	-	-	-	-	100	-	-
NaLIR	33	-	-	90	-	-	100	-	-

This work resulted in a publication at the top data management conference, *ACM* Special Interest Group on Management of Data (SIGMOD)[46]. This publication contains the full evaluation results.

No single system can handle any form of textual query. That points to the need for a hybrid approach. For this purpose, our meta-search system THOR[47] that integrates different NL-to-SQL systems has been built on this observation, and allows INODE to combine the power of different systems, such as SODA, ValueNet, and NaLIR.

We have also started to build a qualitative evaluation of several deep learning techniques for NL-to-SQL systems. Initial results have been presented at EDBT, the 24th International Conference on Extending Database Technology[48], while a more complete study will be presented at ACM SIGMOD[49].

[46] O. Gkini, T. Belmpas, G. Koutrika, Y. Ioannidis. An In-Depth Benchmarking of Text-to-SQL Systems. ACM SIGMOD 2021

[47] T. Belmpas, O. Gkini, G. Koutrika. Analysis of Database Search Systems with THOR. ACM SIGMOD (demo paper), 2020

[48] G. Katsogiannis-Meimarakis, G. Koutrika (2021). Deep Learning Approaches for Text-to-SQL Systems. In *Proc. of the 24th Int. Conference on Extending Database Technology (EDBT).*

[49] G. Katsogiannis-Meimarakis, G. Koutrika (2021). A Deep Dive into Deep Learning Approaches for Text-to-SQL Systems. In *Proc. of ACM SIGMOD*. ACM

4 User Assistance

In this section, we provide a thorough evaluation of the User Assistance components.

4.1 SQL-to-NL: Explaining SQL Queries Using Natural Language with Logos

The evaluation of *Logos*, is divided into two parts: (a) the *automated evaluation* part, where we evaluate our results using established automated metrics, and (b) the *human evaluation* part, where we evaluate our results using the help of SQL experts. The purpose of the first part is to use well-established metrics to show how good the NL explanations are, while the second part aims at evaluating qualitative aspects of the NL explanations, such as clarity and fluency.

Our goal is to investigate how the updated version of Logos (see deliverable D3.2), denoted as *logos v.2,* leads to better translations than those obtained by the previous version, denoted as *logos v.1*. Moreover, we want to know how close to the *ground truth* (textual explanations given by members of the project) the system's explanations are (both versions).

4.1.1 Test Queries

For both types of evaluation, we used the same queries. 28 queries were created (14 for the CORDIS database, and 14 for the SDSS database), all capturing the new features of Logos (see Table 4.1 and Table 4.2, respectively). For more details about the new features of Logos, see deliverable D3.2.

Table 4.1: *The 14 SQL queries of the CORDIS database.*

Id	CORDIS SQL queries
1	`SELECT sum(pm.ec_contribution) AS funding_received FROM projects p, project_members pm WHERE pm.project=p.unics_id AND p.framework_program='H2020';`
2	`SELECT c.name FROM institutions i, countries c WHERE c.unics_id=i.country_id AND i.name='Athena';`
3	`SELECT pr.title FROM projects pr, project_subject_areas psa, subject_areas sa WHERE pr.unics_id = psa.project AND psa.subject_area = sa.code AND sa.title = 'Mathematics and Statistics';`
4	`SELECT distinct t.title FROM projects pr, project_topics pt, topics t WHERE pr.unics_id = pt.project AND pt.topic = t.code AND pr.end_year = 2014;`
5	`SELECT p.full_name FROM people p, projects pr, project_topics pt, topics t WHERE p.unics_id = pr.principal_investigator AND pr.unics_id = pt.project AND pt.topic = t.code AND t.title = 'Systems';`

6	SELECT m.title FROM people p, projects pr, project_programmes pm, programmes m WHERE p.unics_id = pr.principal_investigator AND pr.unics_id = pm.project AND pm.programme = m.code AND p.full_name = 'Thomas Bell';
7	SELECT p.acronym FROM projects p, project_members pm, institutions i, countries c WHERE p.unics_id = pm.project AND pm.institution_id = i.unics_id AND i.country_id = c.unics_id AND c.country_name = 'Greece';
8	SELECT pe.full_name FROM projects pr, people pe WHERE pr.principal_investigator = pe.unics_id AND pr.start_year = 2014;
9	SELECT i.institutions_name FROM institutions i, countries c WHERE i.country_id = c.unics_id AND c.country_name = 'France';
10	SELECT mb.member_name FROM project_members mb, activity_types a WHERE a.code = mb.activity_type AND a.description = 'Research Organisations';
11	SELECT count(p.title) FROM projects p GROUP BY p.start_year;
12	SELECT count(i.name) FROM institutions i, countries c WHERE i.country_id=c.unics_id GROUP BY c.name;
13	SELECT title FROM topics WHERE title like '%climate%';
14	SELECT count(p.title) FROM projects p WHERE p.start_year=2018;

Table 4.2: *The 14 SQL queries of the SDSS database.*

Id	SDSS SQL queries
1	SELECT objid FROM photoobj WHERE clean=1;
2	SELECT specobjid, z FROM specobj WHERE class = 'QSO' AND zwarning = 0;
3	SELECT objid FROM photoobj WHERE ra > 185 AND ra < 185.1 AND dec < 5;
4	SELECT specobjid FROM specobj WHERE survey = 'segue2';
5	SELECT specobjid FROM specobj WHERE class = 'STAR' AND zwarning = 0;
6	SELECT s.specobjid FROM specobj as s WHERE s.subclass = 'STARFORMING' AND s.class= 'GALAXY';

7	`SELECT s.specobjid FROM specobj as s WHERE s.subclass = 'OB' AND s.class= 'STAR';`
8	`SELECT * FROM photoobj WHERE ra > 100 and dec < 100 AND type = 3;`
9	`SELECT DISTINCT type FROM photoobj;`
10	`SELECT class, count(*) FROM specobj GROUP BY class;`
11	`SELECT g.* FROM specobj s, galspecline g WHERE s.specobjid = g.specobjid and ra < 185 AND dec <25;`
12	`SELECT n.* FROM neighbors n, photoobj p WHERE p.objid = n.objid AND p.b = 1.072 and p.l = 174.535;`
13	`SELECT s.* FROM specobj s, galspecline g WHERE s.specobjid = g.specobjid;`
14	`SELECT p.u, p.g, p.r, p.i, p.z FROM specobj s, photoobj p WHERE s.bestobjid = p.objid AND s.class = 'QSO';`

4.1.1 Automated Evaluation

Automated evaluation was carried out to compare the *generated explanations* (those obtained from logos v.1 - v.2) to the ground truth i.e., textual explanations of SQL queries given by members of the project. The quality of the results is measured using the BLEU metric score[50]. BLEU, or the *Bilingual Evaluation Understudy*, is a score for comparing a candidate translation of text to one or more reference translations. BLEU scores range from 0-100%. A score of 100% means that the estimated, by the system, explanation matches completely the ground truth.

The results are summarized in Table 4.3. We also report the minimum and the maximum BLEU score. Furthermore, we noticed a large variation between the scores; thus, we decided to report the median BLEU score per translation system instead of the average. For both databases, median scores are under 10%. Looking at the medians, we conclude that logos v.2 produces translations closer to the ground truth than those obtained from logos v.1. Especially for the CORDIS database, the median bleu score of logos v.2 is more than two times higher than that of logos v.1.

The low scores do not indicate that the translations produced by Logos are not correct. BLEU scores work by counting matching n-grams in the candidate translation to n-grams in the reference text, where 1-gram or unigram would be each token and a bigram comparison would be each word pair. That means that the score is higher the more common parts a NL explanation has with the ground truth. Manual examination of the NL explanations that the two versions of Logos generated versus the ground truth showed that the automatically

[50] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proc. of the 40th annual meeting of the Association for Computational Linguistics* (pp. 311-318).

generated translations were in fact correct. However, they looked very different from the ground truth. Indicatively, we show the translations of query 9.

CORDIS query 9 explanations:

- logos v.1: "*Find the institutions names of institutions associated with countries whose country name is France.*"
- logos v.2: "*Find institutions located in countries whose name is France.*"
- ground truth: "*Show names of institutions from France.*"

We see that our system would not necessarily produce translations the way that a human mind would produce. And even different people would provide different explanations for the same SQL query (albeit all correct). These observations lead to the need of conducting human evaluation as well, which will be presented in the next subsection. They also show the opportunity of enhancing the translation capabilities of the system with learning that not only leverages the database schema but is also performed on previously defined human translations.

Focusing now on the results of CORDIS (see Table 4.3), we see that there is significant improvement on the translations of queries with id 3-6, 11, and 12 (see Table 4.1). This is mainly due to the exclusion of bridge tables, i.e. tables storing foreign keys, from the translation procedure, and the heading attribute addition, i.e. the attribute that represents its relation best (see deliverable D3.2).

For queries with id 9, and 10 we noticed a score reduction. Indicatively, looking at the translations of query 9 above, we observe that although the translation of logos v.2 is more natural than that of logos v.1, the presence of the sentence "*names of institutions*" in the translation of the latter leads to a higher BLEU score.

Let us now focus on the results of SDSS (see Table 4.3). The scores are lower than those of the CORDIS database. This is due to the nature of the SDSS database that uses abbreviated names and letter symbols in order to describe the content of its tables and attributes. For instance, "*photoobj*" instead of "*photometric objects*", or the letter "*u*" to denote the magnitude of a photometric object in "*u*" (ultraviolet) filter. During the experiments, we realized that by transforming those names and symbols into meaningful textual sentences (annotated database graph, see deliverable D3.2), we increase the size of the explanations compared to the size of explanations provided by the astrophysicist expert. This shows another challenge for the automatic generation of NL explanations: different styles of explanations may be given by domain experts in different fields. For example, for query 14 (see Table 4.2), the BLEU score of logos v.2 is substantially lower than that of logos v.1. The translations of query 14 are the following.

SDSS query 14 explanations:

- logos v.1: "*Find the u, g, r, i and z of photoobj associated with specobj whose class is QSO.*".

- logos v.2: "*Find the magnitude u, magnitude g, magnitude r, magnitude i and magnitude z of photometric objects corresponding to spectroscopic objects whose class is QSO.*".
- ground truth: "*Show me the u, g, r, i, z magnitudes of spectroscopic quasars.*".

We concluded that this kind of notation (abbreviated names, and letter symbols), for the attributes of the SDSS tables, is sometimes preferred over full descriptions.

Lastly, it has been observed that the SDSS database includes many discrete variables (attributes) that define different types of objects, e.g. stars. Currently, Logos is incapable of understanding the possible values of an attribute. For instance, it cannot recognize that "*type = 3*" means photometric objects that are galaxies[51]. This justifies the low bleu scores in both versions of our system. A fine example of that case is that of query 7 (see Table 4.2).

SDSS query 7 explanations:

- logos v.1: "*Find the specobjids of specobj whose subclass is OB and class is STAR.*".
- logos v.2: "*Find spectroscopic objects whose spectroscopic subclass is OB and*
- *class is STAR.*".
- ground truth: "*Find all spectroscopic stars which are massive and hot.*".

We see that both versions of Logos do not understand that "*subclass = OB*" and "*class = STAR*" means massive and hot stars.

[51] Note that the explicit information that type 3 corresponds to photometric objects is not stored directly in the database but in the manually enriched ontology (virtual knowledge graph). However, currently Logos only uses information stored in the database schema and does not consider the ontology. Considering also the ontology when translating SQL to NL is part of future work.

Table 4.3: *BLEU scores for the textual explanations of the 14 CORDIS queries, and the 14 SDSS queries.*

Query ID	*CORDIS BLEU SCORES*		*SDSS BLEU SCORES*	
	Logos v.1	Logos v.2	Logos v.1	Logos v.2
1	3.21	3.98	4.99	12.55
2	4.37	6.27	2.45	2.66
3	2.26	9.26	11.71	15.73
4	3.51	12.87	4.86	5.01
5	2.01	15.46	2.84	3.67
6	3.40	11.20	3.38	4.07
7	4.03	4.32	3.74	4.37
8	8.23	9.55	14.72	14.01
9	12.30	9.29	4.46	18.80
10	18.30	7.41	4.03	4.07
11	14.46	24.81	18.46	27.36
12	10.70	17.40	6.87	8.56
13	4.07	4.20	4.30	22.24
14	3.67	4.46	35.61	5.50
MAX	18.30	24.81	35.61	27.36
MIN	2.01	4.20	2.45	2.66
MEDIAN	4.05	9.28	4.66	7.03

4.1.2 Human Evaluation

For this experimental setting, an online survey was conducted. A total of 21 people, all SQL experts, participated in the survey.

From a pool of 28 SQL queries (see Table 4.1, and Table 4.2), participants were asked to rate the textual explanations of 4 randomly chosen queries (2 per database). The queries were equally distributed to all participants. For each query the participant rated 3 explanations (1 per translation system): (a) the explanation produced by logos v.1, (b) the explanation produced by logos v.2, and (c) the ground truth explanation, resulting in a total of 84 explanations rated by humans.

The participants judged the quality of the translations on the seven-point Likert-scales. These scales measures:

- feature *clarity*: how clear and understandable the explanation is
- feature *fluency*: how natural the explanation is
- feature *precision*: how well the information of the provided SQL query is captured on its textual explanation.

Data associated with respondents that completed the survey in less than 5 minutes (half the approximate time for filling out the survey) were deleted. Furthemore, we deleted the data of participants which have selected the same response to every question, regardless of the question. After cleaning the data, we ended up having 2 different scores (per explanation, and feature), corresponding to 2 different participants. The *final score* of an explanation for a given feature is obtained by taking the average of the 2 different scores. Therefore, we ended up having 1 single score per explanation, and feature. Indicatively, in Table 4.4 we show the data collected for the explanations of the CORDIS query with id 9, and the obtained final scores.

From those final scores, 18 rating sets (2 databases x 3 translation systems x 3 features) consisting of 14 elements each (1 for every query), were created.

Table 4.4: *Final scores per feature for the explanations of the CORDIS query with id 9.*

Features	*Rating*	*CORDIS QUERY ID 9*		
		Logos v.1	Logos v.2	Ground Truth
Clarity	Expert A	2	6	7
	Expert B	4	6	7
	Final Score	3	6	7
Fluency	Expert A	1	3	7
	Expert B	4	4	7
	Final Score	2.5	3.5	7
	Expert A	7	7	7

Precision	Expert B	6	7	7
	Final Score	6.5	7	7

In Table 4.5, we show the averages of those sets, accompanied with their standard deviation between brackets. Apparently, logos v.2 leads to better translations in terms of clarity and fluency for both databases (average score increases). However, we see that these scores do not surpass those of the ground truth. An interesting observation is that as the explanations become clearer and more fluent, precision decreases. In other words, as the explanations become more natural, they tend to lose their ability to explicitly explain each part of their associated SQL query.

Lastly, we see that the difference between the average fluency scores of logos v.2 and logos v.1, increases for the SDSS database. As mentioned in the previous section, this is due to the nature of the SDSS database which has a less explainable database schema in terms of natural language explanation. By adopting labels for the components of the database schema (tables, attributes, and joins), we increase the average fluency score (logos v.2).

Table 4.5: *Average clarity, fluency, and precision scores per database, and translation system, along with their standard deviation between brackets.*

Features	*CORDIS*			*SDSS*		
	Logos v.1	Logos v.2	Ground Truth	Logos v.1	Logos v.2	Ground Truth
Clarity	4.25 (1.29)	5.79 (1.13)	6.79 (0.31)	4.32 (1.11)	6.04 (0.77)	6.50 (0.57)
Fluency	3.64 (1.43)	4.75 (1.06)	6.86 (0.35)	3.39 (0.97)	5.50 (0.87)	6.57 (0.56)
Precision	6.04 (1.46)	5.79 (1.10)	5.18 (1.75)	6.50 (0.50)	6.21 (0.70)	5.18 (1.01)

4.2 Query Recommendations with PyExplore

PyExplore aims to provide useful SQL recommendation given an initial SQL query.

PyExplore produces SQL query recommendation by adding a WHERE-clause to the initial query if there was no WHERE-condition in the initial query or by augmenting the WHERE-clause with new conditions. PyExplore performs query recommendations in two stages:

1. In the first stage, PyExplore performs *dimensionality reduction* in order to address the curse of dimensionality. To do this, PyExplore creates *views* (meaning groups of attributes). This way instead of having the complete n-dimensional dataset, we have a set of views of lower dimensionality. To create those views, PyExplore computes the correlation between all attributes of the dataset and groups together highly correlated attributes up to the *maximum number of attributes per view* provided by the user. In order to group together correlated attributes, PyExplore uses *hierarchical clustering*.

2. In the second stage, for each subset of attributes (view) identified by the first step, PyExplore clusters the initial query results using the values of the attributes in the subset. For each subset, the resulting cluster labels are fed into a decision tree classifier to produce the split points of the data. The resulting split points are used to create the recommended SQL queries. To sum up, PyExplore produces *k* SQL query recommendations (*query completions*) for each of the produced views.

For example, consider the following running query on the CORDIS dataset.

CORDIS query example:

```
SELECT total_cost, ec_max_contribution, framework_program,
ec_fund_scheme FROM projects;
```

PyExplore finds that framework_programe and ec_fund_scheme are correlated and form a view. Then, the recommended queries propose meaningful values for the framework_programe and ec_fund_scheme such as “FP7” and “H2020”.

In our experiments, we evaluate pyExplore in terms of: (a) *execution time* and (b) *quality of recommendation* on the CORDIS dataset and the SDSS dataset.

In terms of execution time we measure the time in seconds for the two stages of pyExplore:

1. The time to *generate the views* by computing the correlation between the attributes of the dataset.
2. The time to generate the query recommendations using clustering and the decision tree classifier.

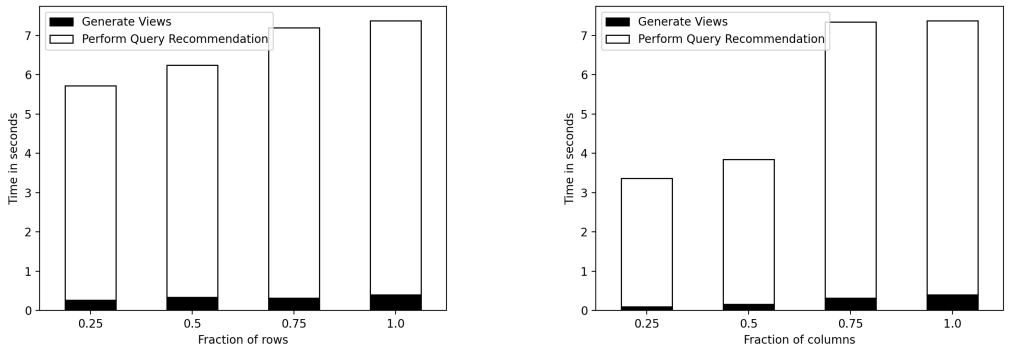
In terms of quality of recommendations, we measure the density of the produced clusters (higher density is better and the possible range is between 0.0 and 1.0).

For our experiments, we calculated the recommendation score for a varying number of recommendations and varying number of attributes per view. For the number of query recommendations (query completions), we used values 2, 4 and 8. For the maximum number of attributes we used the values 3, 6 and 8 for CORDIS and 3, 6 and 10 for SDSS.

Results for the CORDIS dataset

For the CORDIS dataset we used the initial query "`SELECT * FROM projects`". This query for CORDIS produces a dataset of 50,823 rows and 8 columns.

When measuring *execution time*, we performed two different experiments, one *sampling rows* and one *sampling columns* to determine the effect of the number of rows and the number of attributes on execution time. We perform sampling after loading the dataset in memory using the sampling functionality provided by Pandas Dataframes.



(a) Varying the number of rows (b) Varying the number of columns

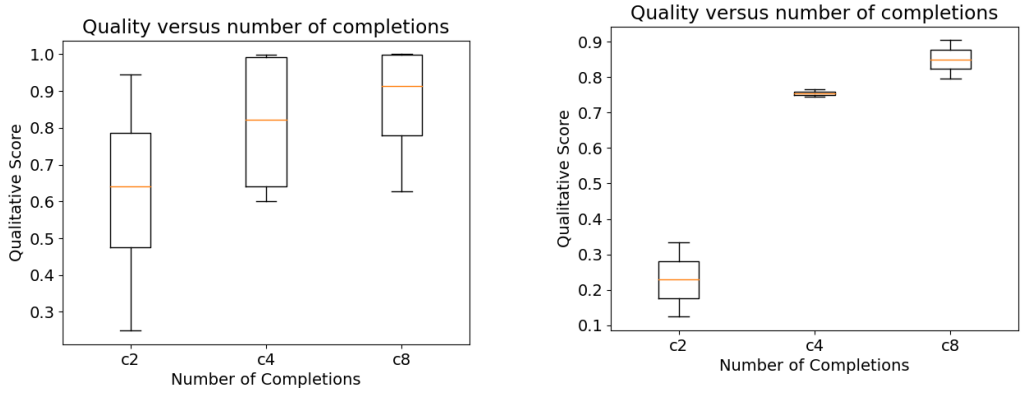
Figure 4.1: *Execution time with (a) the number of rows and (b) the number of attributes of the CORDIS table "projects".*

In Figure 4.1(a), we see the execution times for 25, 50, 75 and 100% of the rows for the CORDIS dataset. We can see that the execution time slowly changes with the number of rows (sublinear relationship), which is a good sign. On the other hand, in Figure 4.1(b), we see that the number of attributes has a direct impact on the performance. In particular, we can see that sampling the number of columns can lead to a reduction in execution time.

To sum up, although we would expect a linear reduction in execution time when using sampling, this was not the case for the CORDIS dataset probably because it is a relatively small dataset.

Next, we are going to examine the *quality of recommendations* for the CORDIS dataset. Following previous approaches[52], we measure the *density* of the produced clusters. Since each cluster maps to a query completion, in this way, we capture an objective measure of the recommendation quality. The possible range for density values is between 0.0 and 1.0. Higher density is better signifying that the clustering of the initial query results is of high quality. Figure 4.2 shows the quality for varying number of completions (c=2, 4, 8). Figure 4.2(a) uses the maximum number of attributes per view, i.e. 3, whereas Figure 4.2(b) uses the maximum number of attributes per view, i.e. 6.

[52] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. ACM Comput. Surv., 38(3):9–es, Sept. 2006



(a) Maximum number of attributes per view is equal to 3.

(b) Maximum number of attributes per view is equal to 6.

Figure 4.2: *Quality for various number of completions (CORDIS dataset).*

In Figure 4.2(a), we can see that setting the maximum *number of attributes per view equal to 3* gives the best results especially for low numbers of completions compared to a larger maximum number of attributes per view. This makes sense intuitively since dimensionality reduction helps us produce relatively dense clusters even for low numbers of completions. In Figure 4.2(b), we can see that by increasing the maximum attributes per view the scores for low numbers of completions start to degrade.

In Figure 4.3, we see how quality is impacted when no dimensionality reduction is used since the maximum number of attributes per view is equal to the number of attributes in the table thus returning a single view. Here we can see that a number of completions less than 8 produces a very low score.

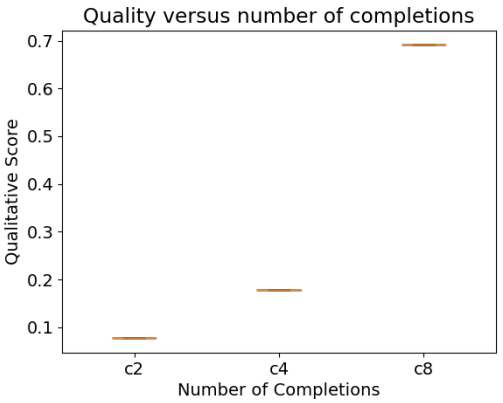


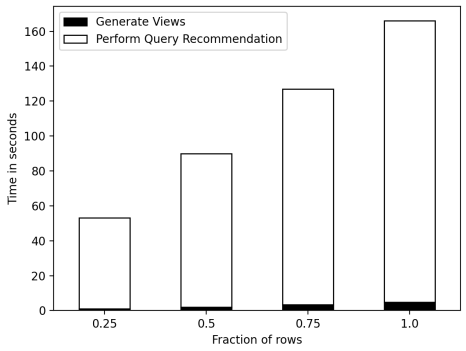
Figure 4.3: *Quality for various numbers of completions with maximum number of attributes per view equal to 8.*

Figures 4.2 and 4.3 reaffirm our intuition that dimensionality reduction is useful even for tables with a relatively low number of attributes.

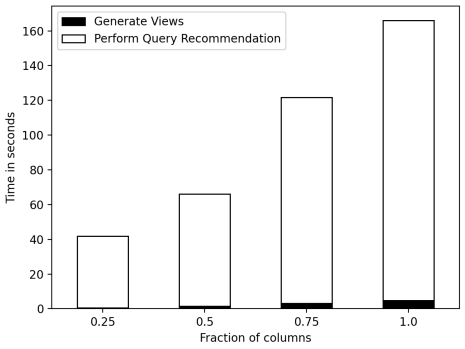
Results for the SDSS dataset

The sample SDSS dataset has one table that consists of 2,616,450 rows and 10 columns. We repeat the same series of experiments as we described above for CORDIS.

Figure 4.4 shows the *execution times* with varying the number of rows (see Figure 4.4(a)) and the number of columns (see Figure 4.4(b)). In Figure 4.4.(a), we see that sampling rows leads to an almost linear decrease in execution time. In Figure 4.4(b), we see the execution time for 25%, 50%, 75%, and 100% of the columns for the SDSS dataset. We see that sampling columns leads to linear decrease in execution time. To sum up, sampling rows or columns leads to linear decrease in execution time.

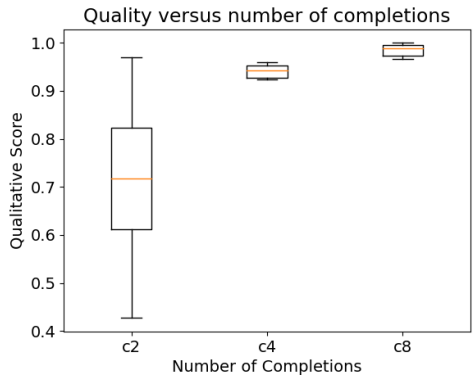


(a) Varying number of rows.

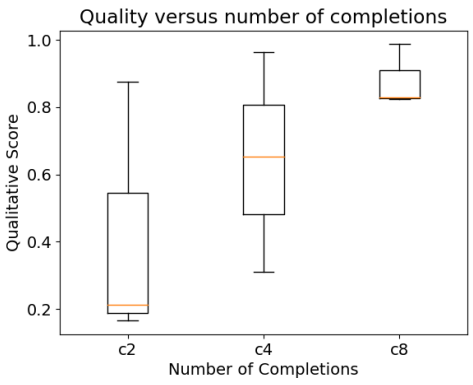


(b) Varying number of columns.

Figure 4.4: *Execution time with (a) the number of rows and (b) the number of attributes of the selected SDDS table.*



(a) Maximum number of attributes per view is equal to 3.



(b) *Maximum number of attributes per view is equal to 6.*

Figure 4.5: *Quality for various number of completions (SDSS dataset).*

Next we are going to examine the *quality of recommendations* for the SDSS case. Figure 4.5(a) shows that setting the maximum number of attributes per view to 3 gives the best

results especially for low numbers of completions. This makes sense intuitively since because of the dimensionality reduction we can produce relatively dense clusters even for low numbers of completions. In Figure 4.5(b), we can see that by increasing the maximum number of attributes per view, the scores for low numbers of completions start to degrade. For larger values of the number of completions (for example for completions equal to 8) the difference is smaller. However, for completions equal to 4 the scores show a much larger range. Especially for completions equal to 2 the effect of the larger number of attributes is clear.

Finally, Figure 4.6 shows how quality is impacted when no dimensionality reduction is used since the maximum number of attributes per view is equal to the number of attributes in the table thus returning a single view. As we can see, the score for all numbers of completions is significantly lower compared to the case with dimensionality reduction. We see that dimensionality reduction is indeed very important to produce useful recommendations especially for lower numbers of completions.

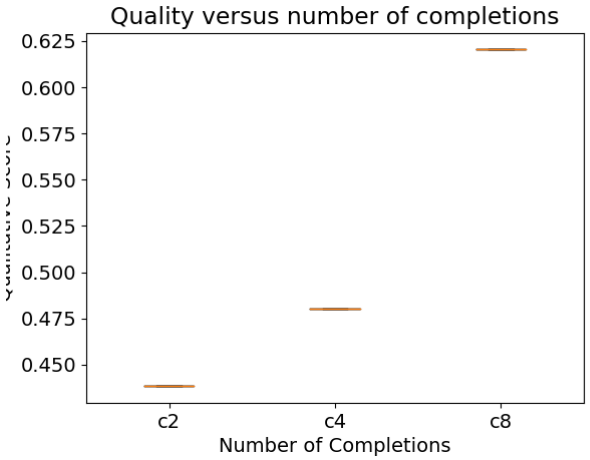


Figure 4.6: *Quality for various number of completions with maximum number of attributes per view is equal to 10 for the SDSS dataset.*

5 Multi-Modal Discovery

The objective of Multi-Modal Discovery services is to enable the exploration of search results and the interactive query manipulation that go beyond lists of results, and text input fields for querying. It is looking for visualizations that put queries, search results and user session history in context, emphasizing relations between items of interest, so that the user yields clues for proceeding with the exploratory search and deciding whether the exploration has finished.

In this section, we report the progress on the three tasks that are part of Work Package 7:

1. Task 7.1 Visual guidance and exploration of search results
2. Task 7.2 Interactive manipulation and optimization of queries
3. Task 7.3 Integrated seamless query-response loop

Before diving into the details of this section, a few terms have to be defined. From a high-level perspective, OpenDataDialog is a search engine: Users issue queries, and the system responds with several candidate results. A query is the input which is fed into the system. Technically, INODE provides two types of input currently: a query string input (used by NL-to-SQL and NL-to-SPARQL) and structured query inputs (SQL and SPARQL queries, By-Example, By-Recommendation, SQL-to-NL), although other types of input may be possible (see Section 5.5.1). Candidate results are the output of the search engine, and each candidate result is represented by one table.

Please note that we follow the definition of Zhang and Balog[53] which is closer to tables in the sense of spreadsheets than in the sense of database schema definitions. Tables consist of headings, columns, rows and entities (see Figure 5.1). In this section, “spreadsheet”, “result”, “candidate”, and “result set” are used as synonyms for tables.

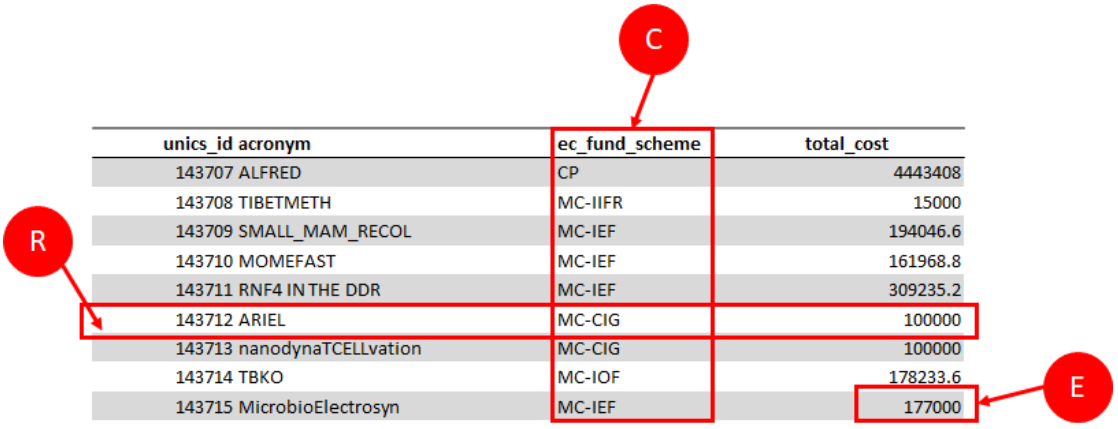


Figure 5.1: *Elements of a table as used in this section. C denotes columns, E denotes entities (or cells), R denotes rows.*

[53] S. Zhang and K. Balog, “Web Table Extraction, Retrieval, and Augmentation: A Survey,” *ACM Trans. Intell. Syst. Technol.*, vol. 11, no. 2, pp. 1–35, Mar. 2020, doi: 10.1145/3372117.

In the following, we describe the task analysis (Section 5.1) and the qualitative survey to derive a prioritized work plan (Section 5.2) that we conducted. Those influenced the design of the visual results exploration module (Section 5.3) and the guidance and orientation module. Work on guidance has been split into a visual part (Section 5.4) and a technical part (Section 5.5). Lastly, we summarize our progress in Section 5.5.

5.1 Task Analysis

The goal of most search engines is to retrieve the information that satisfies the users' need for information best. In OpenDataDialog, the goal of the users is to find the table that best suits their need for information. To accomplish that, the users must make various decisions, for example:

1. Decide to *restart the process* by issuing another query, for example, if the user can see at a glance, that all results are inapt.
2. Decide to investigate the current candidate result set.
3. Decide if a single *candidate table is worth* to be taken into consideration for further investigation.
4. Decide which *table matches the visceral need* for information best, when comparing two or more candidate tables.
5. Decide if any candidate *table satisfies the information need*, and whether the exploration process is finished.
6. Decide to *apply a downstream operator* on a table, column, row of cell and continue the exploration process.

Multi-Modal Discovery services focus on decisions 3 and 4, where the user decides between the candidate options, as this is the decision where information visualization can support the user best. The comparison of tables can be approached from different angles. The classic approach is the metadata catalog, which, when applied to data sets, records various metadata about the origin of the data, its authors, purpose, date and time of creation, etc. Most data set search systems, such as the European Data Portal[54], work on this level. However, some use cases simply cannot be answered on this level, i.e., users cannot decide based on metadata alone. These users need to gain insight into the data set to evaluate its use for their case. For example, if they require a data set to contain a specific information, such as

- the resolution for digital cameras
- the electric drive range capacity for cars
- the “EU framework programme codes” for research projects
- the subclass of stellar spectral objects.

To the best of our knowledge, no data set search engine provides enough information directly on the result overview page to decide. The user has to drill down to the individual data set in order to be able to check that. Even with the inclusion of structural metadata[55],

[54] https://data.europa.eu/

[55] data set size information such as number of rows, columns, numerical and categorical cells, as well as data type information like percentages of null and unique values

like “Loch Prospector”[56] proposes, it does not provide an *overview* over the result space. With the Multi-Table Explorer, we move beyond this, and *integrate statistical properties of the data directly into the result view*. Details about the look and feel are reported in greater detail in deliverable D3.2.

5.2 Prioritizing and Planning

Deliverable D2.1 provides a comprehensive list of requirements. However, to infer an actionable implementation plan from it which mitigates the risk to address the wrong problems or to choose the wrong abstraction, it was important to translate the requirements into a prioritized list of features[57]. We briefly describe the questionnaire and its answers before we present the prioritized feature list for Multi-Modal Discovery services.

We conducted a survey with a total of 66 questions, from which 52 were five-point Likert-scale questions. Many questions originate from our task analysis and deliverable D2.1. Analysis of agreement and variance across user groups (a) helped to derive a prioritization of features, (b) to set up a work package implementation plan, and (c) to inform the design of the visual exploration interface. Questions that were answered with low variance and high consensus have been considered for baseline requirements while answers with high variance, for example due to strong domain focus, are taken into account for use case specific enhancements. For convenience, we list it, alongside their current status, in Table 5.1.

Our use case partners MPI, SIB, and SIRIS identified seven user groups that can be summarized as follows. Users confirmed our assumption that query construction is an iterative process in which the final query is composed from multiple smaller ones. They also strongly agreed that, in an iterative setting, it is desirable to highlight differences between a predecessor and its successor exploration steps (for both, query and result). In general, users are results-oriented and prefer to operate on result tables rather than queries. This is also reflected in their agreement on direct table manipulation as a method to formulate queries. Other means of issuing queries, such as faceted search, search forms and node-link diagrams had less agreement and thus got assigned a lower priority. Regarding the visualization of results, all user groups are at least familiar with spreadsheets. Relevant data visualizations are bar charts, histograms, pie charts and line charts. Domain specific diagram types have also been collected, namely color-color diagrams, sky observation images, and geospatial maps.

[56] N. Makhija, M. Jain, N. Tziavelis, L. D. Rocco, S. D. Bartolomeo, and C. Dunne, “Loch Prospector: Metadata Visualization for Lakes of Open Data,” presented at the 2020 IEEE Visualization Conference (VIS), Oct. 2020, doi: 10.1109/VIS47514.2020.00032.

[57] T. Munzner, “A Nested Model for Visualization Design and Validation,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 921–928, Nov. 2009, doi: 10.1109/TVCG.2009.111.

Table 5.1: *Overview over features identified for Multi-Modal discovery services. Sorted by task, priority and status. Status as of 2021-April-30.*

id	Task	Feature description	Priority[58]	Source[59]	Status[60]
1	T7.1	Enable visual comparison of tables	MUST	TA	Done
2	T7.1	Enable inspection of table result details	MUST	TA	Done
3	T7.1	Visualize result tables as spreadsheets	MUST	Q	Done
4	T7.1	Basic diagrams like bar chart, histograms, pie charts, line charts	MUST	Q	Done
6	T7.1	Include user guidance to support the visual exploration of the search space	MUST	D2.1	Ongoing
12	T7.1	Assistance to broaden and narrow the search space	SHOULD	Q	Done
13	T7.1	Display sky observation images	SHOULD	Q	Done
14	T7.1	Assistance to broaden and narrow the search space is preferable (covered by WP5)	SHOULD	Q	Done
15	T7.1	Enable the exploration of search results	SHOULD	D2.1	Ongoing
16	T7.1	Visualizations that emphasize relations between items of interest, so that the user yields clues for proceeding with the exploratory search and deciding whether the exploration has finished.	SHOULD	D2.1	Ongoing
18	T7.1	Visualizations that put queries, search results and user session history in context	SHOULD	D2.1	Planned
19	T7.1	Color-color diagrams and other use-case specific visualizations	SHOULD	Q	-
20	T7.1	Highlight differences between predecessor and successor query	SHOULD	Q	-
25	T7.1	Visualize geographic location for CORDIS database	MAY	Q	-
29	T7.1	Enable users to memorize search results during exploration	MAY	D2.1	-
30	T7.1	Enable users to arrange search results during exploration	MAY	D2.1	-
31	T7.1	Enable users to annotate search results during exploration	MAY	D2.1	-
32	T7.1	Enable users to structure search results during exploration	MAY	D2.1	-
7	T7.2	Enable users to interactively specify queries	MUST	D2.1	Ongoing
8	T7.2	Enable interactive query manipulation that goes beyond lists of results, and beyond text input fields for querying.	MUST	D2.1	Ongoing
9	T7.2	Enable users to interactively manipulate queries	MUST	D2.1	Planned
10	T7.2	Enable direct manipulation mechanisms in the visual result representation	MUST	D2.1	Planned
17	T7.2	Direct table manipulation for query formulation	SHOULD	Q	Ongoing
21	T7.2	Enable users to promote relevant search results.	MAY	D2.1	-
22	T7.2	Enable users to remove irrelevant search results from the visualization	MAY	D2.1	-
23	T7.2	Enable users to rearrange search results visually	MAY	D2.1	-
24	T7.2	Provide an iterative query-response experience, similar to computational notebooks	MAY	Q	-
26	T7.2	Faceted search for search space narrowing/broadening	MAY	Q	-
27	T7.2	Visualize queries as node-link-diagram	MAY	Q	-
5	T7.3	Enable users to use operators on table, row, column, and cell level.	MUST	Q	Done
11	T7.3	Enable users to seamlessly switch between exploration and query manipulation modes	MUST	D2.1	Planned
28	T7.3	Enable users to set up their data model preferences to ease their day-to-day work with schemas	MAY	Q	-
33	T7.3	Generate an engaging experience for the user	MAY	D2.1	-
34	T7.3	Provide user interface variant to support novice users	MAY	D2.1	-
35	T7.3	Provide user interface variant for expert users	MAY	D2.1	-

[58] The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this table are to be interpreted as described in RFC 2119.

[59] TA: Task analysis, Q: Questionnaire, D2.1: Deliverable D2.1

[60] Available statii: - (None), Planned, Ongoing, Done (in that order).

5.3 Exploring Search Results Visually

In essence, the questionnaire reassured that the Multi-Modal Discovery services should focus on results presentation and that the results should be visualized as tables if possible. The main goal of the Multi-Table Explorer is to enable users to *explore and compare the multiple candidate table results* in one comprehensive view (see Figure 5.2 for an example). Its look and feel is described in the Deliverables D7.1 and D3.2 in greater detail. In this deliverable, we provide an informal evaluation of the currently deployed prototype.

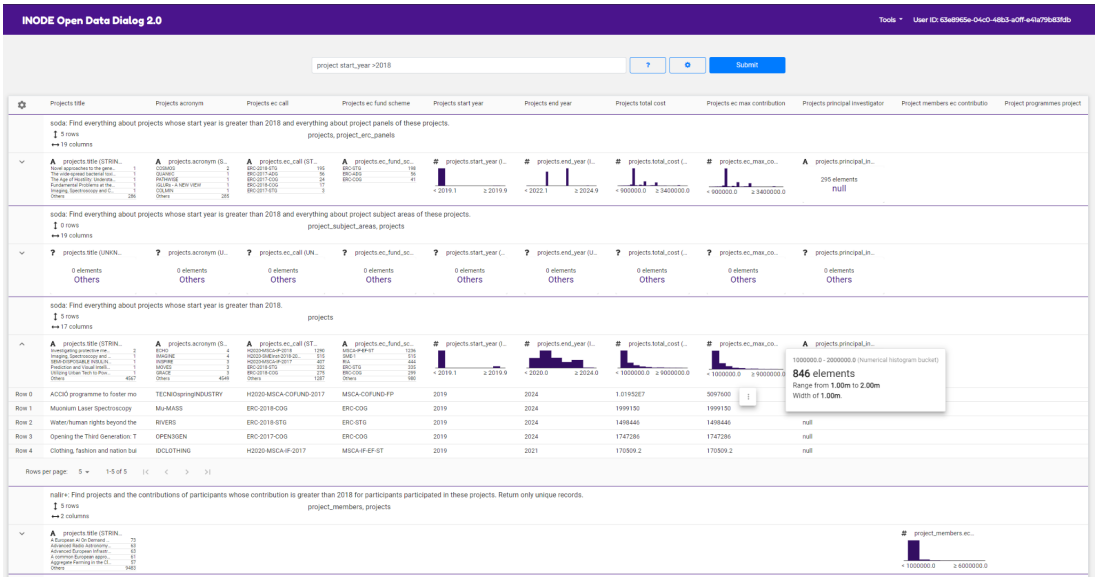


Figure 5.2: *The Multi-Table Explorer provides insight* into *the data. A user looking for projects funded by the European Union can quickly judge that the first and the third result might be most interesting to investigate. Table 3, i.e. the third result, exhibits more interesting diagrams, so that is inspected first. To see an excerpt of the actual tabular data, the user opened the details. Further diving into the third result, they inspect "ec max contribution" by hovering over the diagram with the mouse and see that 846 results had a max contribution between €1.0m and €2.0m.*

User group feedback regarding tabular layout was positive regarding the use of tables as a primary metaphor, but they still felt overwhelmed when the result screen was displayed in its entirety. We attribute this to three causes that are either planned or in progress:

1. *Missing guidance*. The guidance aspect has not yet been integrated into Multi-Table Explorer yet, but is in progress (see also Section 5.4).
2. *Unordered results*. In contrast to general purpose information retrieval systems, the Multi-Table Explorer displays the candidate results in unordered fashion. This is due to the way the OpenDataDialog system works and will be resolved when the embeddings space service is available (see Section 5.5).
3. *Many columns*. The current version of Multi-Table Explorer was designed for roughly 10 data columns, resulting in a screen real estate of 170 pixels per data set column

on Full HD screens[61]. If a result contains more columns, only the 10 most relevant columns are displayed. However, INODE operators also consider data joins between multiple database tables, leading to results which feature more than 100 columns, forcing the user to select columns to hide and show much more often than anticipated. We anticipate to address this in the last year of the project.

Most of the efforts invested in Task 7.1 can be attributed to the Multi-Table Explorer component. Many features have been implemented and enable users to get an overview over the candidate tables that are the result of an OpenDataDialog operator, inspect them in detail and compare candidate tables to reach a decision. To enhance the usability of OpenDataDialog, we implemented the seamless query response loop, enabling the user to trigger operators within the Multi-Table Explorer component already before project month M10, a feature that relates to Task 7.2 and 7.3 which both officially started in project month M13. Ongoing work advances into the direction of guidance and interactive query manipulation.

5.4 Providing User Guidance and Orientation

Another goal of the Multi-Modal Discovery services is to *guide* the user while *navigating the result space*. In this area of ongoing work, we mainly address the features that guide by providing overview and visualizing relations between items of interest. This approach makes use of a high-dimensional vector space to measure distance between search results (aka spreadsheets, or tables) and to retrieve close neighbors of a given spreadsheet. Details about that vector space are given in Section 5.5.

This local approach, named *Result Radar*, addresses both requirements in one view, for a focused set of items of interest. It complements the Multi-Table Explorer by providing a *high-level overview over the candidate results* (see Figure 5.3). Relations between the query and the candidate tables are visualized (query-result axis) as well as the relation among all pairs of candidate tables (result-result axis). That way, the user can see how well candidate tables match to the current query and also at the same time see how tables of the result set relate to each other, thus *gaining the ability to orient in the result space* and hopefully enabling the user to make the right decisions faster.

We have implemented a proof-of-concept (POC) to test if the overall idea of providing orientation is feasible. The POC has limited functionality and uses a small set of 1,000 tables from the Wikitables dataset[62] for testing purposes. But still, it allows one to gain a first impression of the Result Radar.

Users can issue search queries and investigate the query results in a scatter plot. To read the visualization, the user has to learn two things: The better a result matches the query, the closer it is to the center (query-result axis) and the more similar two results are, the likelier they have a similar angle (result-result axis). Once learned, the ResultRadar guides users as

[61] According to https://www.w3schools.com/browsers/browsers_display.asp, more than 50% of users have a screen resolution larger than Full HD (1920x1080).

[62] C. S. Bhagavatula, T. Noraset, and D. Downey, "TabEL: Entity Linking in Web Tables," in *The Semantic Web - ISWC 2015*, Cham, 2015, pp. 425–441, doi: 10.1007/978-3-319-25007-6_25.

regions of similar tables are easily identified. Moreover, the Result Radar also enables the users to judge if the tables within that region match the query well. Thus, the users can decide on the basis of two pieces of information, which table to evaluate next instead of one (the rank).

For example, results in area (C) of Figure 5.3 are similar to each other, but they differ from area (D) in Figure 5.3. The user can navigate and zoom like geomap web applications to investigate dense regions. Also, details can be shown on demand, after clicking an element or after placing a selection rectangle (not depicted) and are a preparation for connecting the Multi-Table Explorer and the Result Radar.

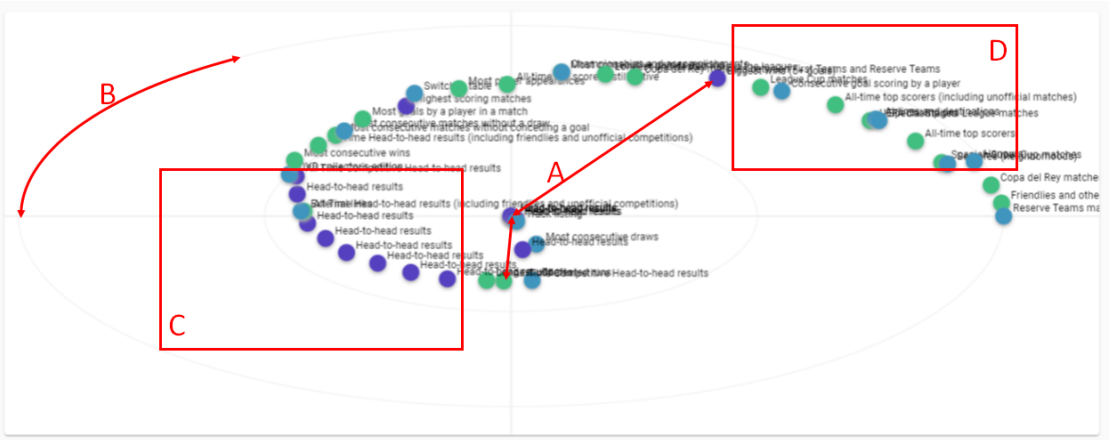


***Figure 5.3**: After searching for "El Clasico"[63], the top 50 candidate results are displayed. (A) The green result is much more relevant than the violet as it is much closer to the center. (B) Results that are similar to each other have similar angles around the center (showing the result-result axis). Points in area (C) are "Head-to-head" comparison tables which are similar to each other. Points in area (D) are similar, mostly related to "top scorers". (C) and (D) are distant from each other, suggesting that "top scorer" tables are dissimilar to "Head-to-head results".*

The Result Radar also exhibits an interesting visualization challenge that can be observed on the 3 o'clock axis of Figure 5.3: The table located "3:01" is maximally dissimilar to the table at "2:59". However, intuitively, similarity should be quite high when items have similar angles. This "disconnectivity" issue is due to the fact how dimensionality reduction algorithms work. Finding a solution to this issue is likely to improve the overall usability and interpretability of this visualization technique. We are currently investigating this and are confident to find a solution for this issue.

As the proof-of-concept puts an emphasis on the radar itself, we also experimented with smaller-sized radar visualizations as can be seen in Figure 5.4. The idea is to evaluate how the Result Radar can complement the Multi-Table Explorer in OpenDataDialog.

[63] https://en.wikipedia.org/wiki/El_Clásico

Isomap[64] MDS[65] PCA[66] t-SNE[67] UMAP[68]

***Figure 5.4**: Showing the top 300 results for the query "El Clasico"[69] using different dimensionality reduction algorithms. Two observations can be made. (a) The radar is sensitive to the choice of the dimensionality reduction algorithm for the result-result axis. (b) The radar enables the user to get an impression of how the search results space is shaped, which regions contain similar tables, and which regions match the query well.*

The visualization approach works well on a small data set with 1,000 tables from the Wikitables dataset[70]. However, we will further improve the visualization model, e.g. to include cross-filtering within the Multi-Table Explorer, to enable users to interactively steer the search with the help of relevance feedback. As the current model is simple and not scalable to thousands of tables, we started on to investigate a more sophisticated approach in Section 5.5.

5.5 Comparing Tables Computationally

The Result Radar requires a method to put tables into relation to each other. A well-established approach to measure the relatedness between items is to define a mapping function which translates each table to a point in an embedding space (a high-dimensional Euclidean vector space) in such a way that similar tables reside at similar locations in that space. The similarity of two tables can then be calculated by computing the distance[71] between vectors. Also, given one table, the most similar tables can be retrieved.

---

[64] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000, doi: 10.1126/science.290.5500.2319.

[65] Multidimensional Scaling - I. Borg and P. Groenen, *Modern Multidimensional Scaling Theory and Applications*. New York, NY: Springer New York, 1997.

[66] Principal Component Analysis - K. P. F.R.S, "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, Nov. 1901, doi: 10.1080/14786440109462720.

[67] t-Distributed Stochastic Neighbour Embedding - L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[68] Uniform Manifold Approximation and Projection - L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction," *arXiv:1802.03426 [cs, stat]*, Sep. 2020, Accessed: Apr. 15, 2021. [Online]. Available: http://arxiv.org/abs/1802.03426.

[69] https://en.wikipedia.org/wiki/El_Clásico

[70] C. S. Bhagavatula, T. Noraset, and D. Downey, "TabEL: Entity Linking in Web Tables," in *The Semantic Web - ISWC 2015*, Cham, 2015, pp. 425–441, doi: 10.1007/978-3-319-25007-6_25.

[71] Deza, M., und Elena Deza. *Encyclopedia of distances*. Fourth edition. Heidelberg: Springer, 2016.

The INODE scenario poses some challenges that have to be overcome and make it particularly challenging to design a table embedding function:

- Related work primarily focuses on tabular data extracted from Wikipedia[72,73,74] or from web crawls[75]. Since those data tables are likely to differ in structure from structured sources like relational databases, we need to investigate the differences so that we can adapt our approaches to learn embeddings accordingly or collect a data set based on structured data sources on our own.
- Using Wikipedia and web crawls as a basis enables related work to include meta-data such as table captions, related documents and other into the inference process. However, in the INODE scenario, this is not always the case, e.g., if users might want to use data from a local spreadsheet application, such information might be unavailable. We assume that this has an impact on the inference process. We therefore have to measure it, and need to address this by focusing on approaches that work on tables without taking contextual information into account if it is too large.
- The astrophysics and the policy making use cases provide plenty of numerical information in their data sets. The current state of the art in table embedding spaces by and large ignores numerical information[76,77]. Including that information into the similarity function should increase the information density of the vector representation and thus improve the accuracy of the similarity measure.

We have implemented a baseline approach based on the Paragraph Vector approach by Le and Mikolov[78]. This model currently provides the basic functionality that is needed to continue work on our visual approaches (Result Radar) while at the same time, we continue to research on learning table embeddings.

---

[72] C. S. Bhagavatula, T. Noraset, and D. Downey, "TabEL: Entity Linking in Web Tables," in *The Semantic Web - ISWC 2015*, Cham, 2015, pp. 425–441, doi: 10.1007/978-3-319-25007-6_25.

[73] L. Deng, S. Zhang, and K. Balog, "Table2Vec: Neural Word and Entity Embeddings for Table Population and Retrieval," *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR'19*, pp. 1029–1032, 2019, doi: 10.1145/3331184.3331333.

[74] B. Fetahu, A. Anand, and M. Koutraki, "TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables," in *The World Wide Web Conference*, New York, NY, USA, May 2019, pp. 2736–2742, doi: 10.1145/3308558.3313629.

[75] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang, "WebTables: exploring the power of tables on the web," *Proc. VLDB Endow.*, vol. 1, no. 1, pp. 538–549, Aug. 2008, doi: 10.14778/1453856.1453916.

[76] L. Deng, S. Zhang, and K. Balog, "Table2Vec: Neural Word and Entity Embeddings for Table Population and Retrieval," In *Proc. of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR'19*, pp. 1029–1032, 2019, doi: 10.1145/3331184.3331333.

[77] B. Fetahu, A. Anand, and M. Koutraki, "TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables," in *Proc. of The World Wide Web Conference*, New York, NY, USA, May 2019, pp. 2736–2742, doi: 10.1145/3308558.3313629.

[78] Q. V. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," arXiv:1405.4053 [cs], May 2014, [Online]. Available: http://arxiv.org/abs/1405.4053.

Given the complexity of the topic, we have prepared an internal tool to support our research. This tool contains two main components, we want to present here: (1) the *Table Editor* and (2) the *Table Map.* The Table Editor can be used to search in the embedding space with minimal information loss. The Table Map visualizes the vector space and allows to drill down into search results and exploration histories.

5.5.1 Table Editor

For evaluating an embedding space, we need a method to flexibly construct arbitrary (but realistic) inputs that can be translated into an embedding vector with the highest possible accuracy. In addition to our internal needs, the users supported the idea to formulate queries by direct table manipulation in the questionnaire as well. In a proof-of-concept, we started to work on a new input modality which allows users to *issue search queries by editing a table*.

We prioritized the implementation of this method over the direct manipulation in the result visualization representation (e.g. dragging points in the radar to provide relevance) for technical reasons. The advantage of the table editor is obvious: We can change the table in a very precise way and thus investigate the behavior of the embedding space very diligently.

The Table Editor is a user interface component, which supports editing tabular data, e.g. by changing a cell, by removing a column, by adding a row, etc. (see Figure 5.7). Besides starting from scratch, the user can also use a previously selected table and adapt it accordingly. Once done, the user can send the table query to the system and inspect the candidate results in the Table Map.

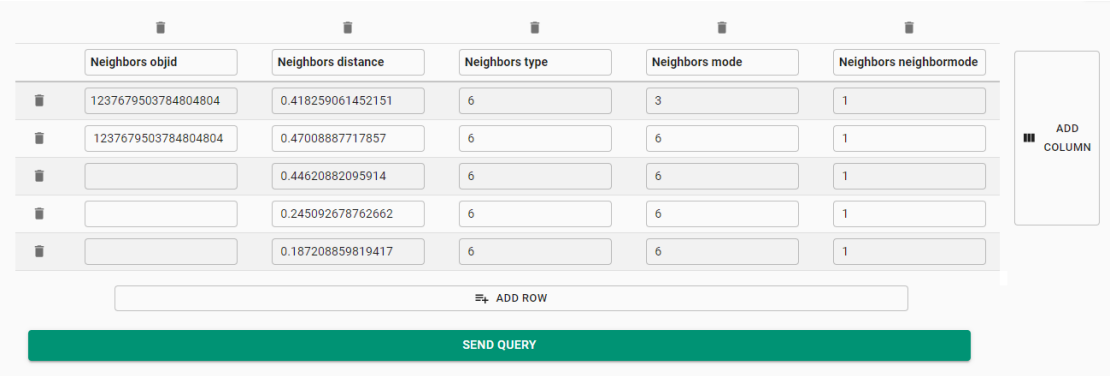


Figure 5.7: *A first experiment for an interactive Table Editor for querying. Based on any given table, or from scratch, the user interactively constructs a table. This table is then used to retrieve similar tables from a database of tables.*

5.5.2 Table Map

The embedding space of tables is huge, easily containing 200 and more dimensions, depending on the model parameters chosen. Making sense of nearest neighbor search is complex. We needed a way to analyze the results of interactive usage of the embedding space, ideally in a reproducible way, so that multiple exploration sessions can be analyzed in parallel.

Global overviews have the advantage that they always look the same (“stable”), and thus can be learned, just like the map of the Earth has been learned in school. Due to the visual

stability, it is a good foundation for visualizing additional information on top of it, for example to visualize the whole exploration history of a user at once (see Figure 5.5) or even to compare multiple user explorations visually (Figure 5.6). As the visualization is precomputed, the approach is very scalable, and allows the user to update visualization at interactive framerates, even when thousands of tables have to be plotted. The Result Radar fails at both of these requirements: It is tightly coupled to 1 single query, and computationally less scalable on-demand processing approach.

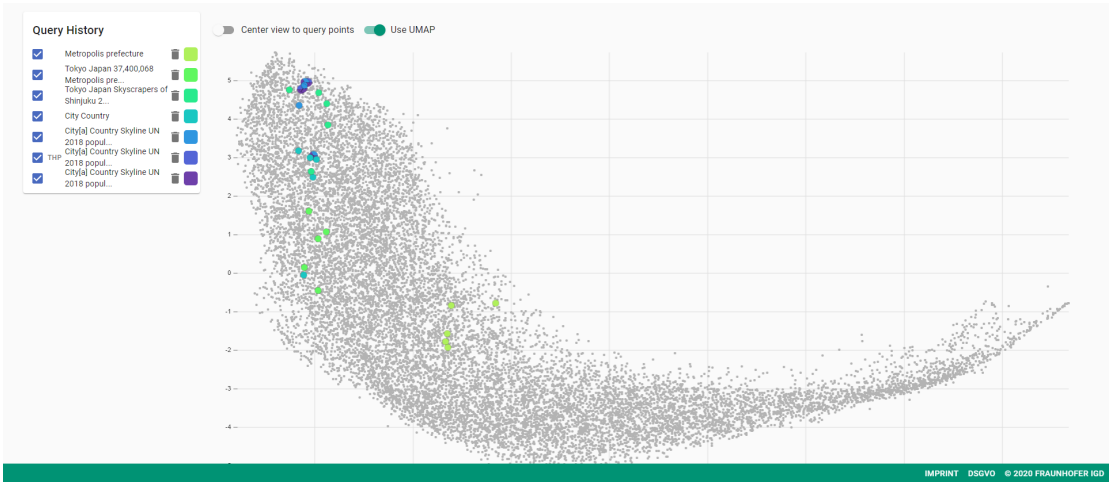


Figure 5.5: *Details about one single exploration session. The sequence of 7 consecutive, interdependent exploration steps is laid over the global table distribution (depicted as grey points, sampled from 60,000 Wikitables) with a violet-green gradient color scale. The user started at the top of the scatter plot (dark blue) and gradually moved downwards (light green). A possible interpretation is that either the initial search results (violet) were not good, so the user decided to dive deeper, or has lost focus and was dragged away.*

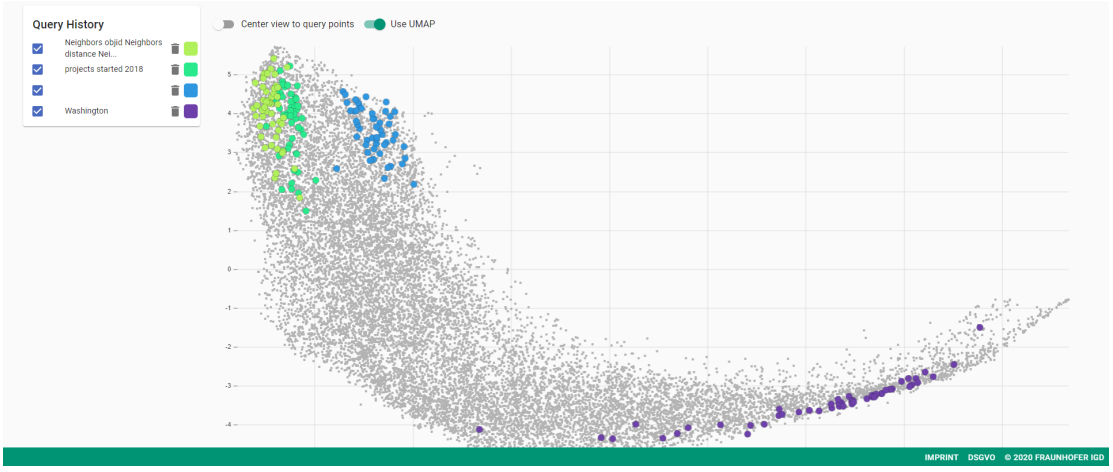


Figure 5.6: *Comparing multiple exploration sessions. Four exploration sessions (with multiple consecutive steps each) are laid over the global table distribution (depicted as grey points, sampled from 60,000 Wikitables), each history with a separate color. A shallow investigation*

suggests that all but violet dealt with queries and tables containing numerical data while violet mainly contained tables with a relation to politics and elections (many names).

The work in this section contributes to the guidance and interactive table manipulation aspects of Task 7.1 and T7.2 and still is in its infancy stages. While a first proof of concept has been developed, more research has to be done. As soon as possible, we will add the similarity service to the set of available INODE components and include it into OpenDataDialog as well.

5.6 Summary

In WP7, we are following the user-centered design approach. We summarized our task analysis and the prioritized work plan for WP7. We presented an informal evaluation of the Multi-Table Explorer (Task 7.1 and Task 7.3) and showed promising initial results of the Result Radar proof-of-concept, which provides orienting guidance during the exploration process (Task. 7.1), and is the basis for interactive query manipulation (Task 7.2). Lastly, we described our efforts in the development of an embedding space for tables (Task 7.2), which included the development of a proof-of-concept for a new query formulation modality.

6 End-to-End Evaluation

The purpose of WP8 (Evaluation work package) is to develop a general framework for evaluating all the components of INODE. We first describe our framework for a single component, which is then applicable to different components of INODE shown in Figure 6.1.

6.1 Our Evaluation Framework

The goal of our evaluation framework is to analyze **Data Factors (DF), System Factors (SF) and Human Factors (HF)** that affect data exploration. Figure 6.1 shows the factors we are measuring in our framework. The proposed factors extend previously proposed work in evaluating data exploration.[79]

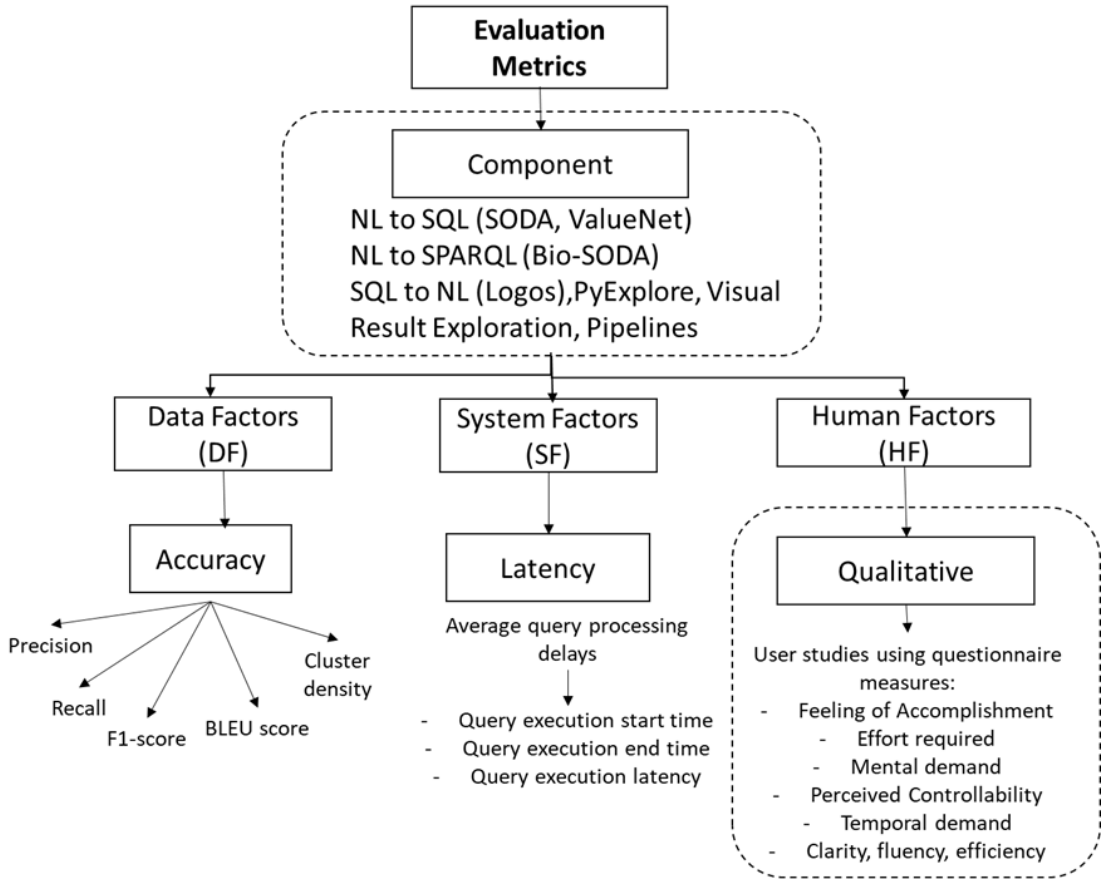


Figure 6.1: *Data, System and Human Factors used in our evaluation framework.*

To measure different factors, we implemented a Logging Mechanism in INODE 2.0, which allows us to log various data, system and user interactions. This enables measuring data

[79]P. Rahman, L. Jiang, A. Nandi (2020), Evaluating interactive data systems. VLDB J. 29(1): 119-146
P. Eichmann, E. Zgraggen, C. Binnig, T Kraska (2018), IDEBench: A Benchmark for Interactive Data Exploration, SIGMOD Conference 2020: 1555-1569

factors, i.e., accuracy in terms of precision and recall, and system factors such as latency for quantitative evaluations. We have presented the values for system factors for different components of INODE (e.g. SODA, Logos, ValueNet) as described in Deliverable D3.2.

6.2 Component Evaluation

Different components of INODE measure a different subset of factors. NL-to-SQL and NL-to-SPARQL sub-components are mostly focused on data and system factors. Data and system factors were described in our previous deliverable.

The component Logos provides user assistance by translating SQL to natural language. The evaluation of Logos is based on both data and human factors. For the former, we used the well established automated metric *BLEU score*, counting the correlation between the human translations (ground truth) and those of our system. In terms of human evaluation, we conducted a survey in which we measured qualitative features of our translations on a seven-point Likert-scale. Those features are: (a) *clarity* (translation's explainability), (b) *fluency* (translation's naturality), and (c) *precision* (translation's precision with respect to the provided SQL query). Details about our experiments can be found in Section 4.

The component PyExplore produces SQL query recommendations given an initial SQL query. The evaluation of PyExplore is based on data. More specifically we measure the density of clusters produced during the recommendation process and use this as a metric for the quality of the produced queries. More details on our evaluation can be found in Section 4.

The Multi-Modal Discovery services provide means to visually assess and explore the result space. Evaluation is currently in preparation and will primarily focus on human factors to assess the suitability of the Multi-Table Explorer to support the decision making process, but may also include system factors. Human factor metrics of interest are task completion time, feeling of accomplishment and as well as the number of interactions. System factors are also taken into consideration to evaluate performance and mainly relate to post-aggregation latency of data processing.

In the following sections of this deliverable, we will describe the qualitative human factors evaluation in the context of using the pipelines component.

6.3 Pipeline Component Evaluation

The purpose of this section is to describe the qualitative evaluation of human factors and showcase its usage for the pipeline sub-component. Although we solely present a benchmark study for galaxy data exploration, the methods showcased are transferable to other datasets.

The pipeline sub-component of INODE is concerned with the evaluation of *fully-guided*, *partially-guided* and *manual data exploration*. This version will focus on evaluating data exploration using manual pipelines - where a user chooses the next operation for data exploration.

We showcase the pipelines sub-component for evaluation of the *Galaxy Data Exploration (GDE)* toolkit that allows its users to explore the SDSS[80] data by choosing from a set of

[80] https://www.sdss.org/

by-example operators designed in WP5. Figure 6.2 represents the various modules of the GDE system that users can interact with.

- Module (1) "**Current pipeline**" consists of a summary of operators utilized by the users to generate the current data subset or simply bins.
- Module (2) "**Unique ID**" represents the unique ID assigned to the user.
- Module (3) "**Current operator results**" represents the collection of a data subset and their corresponding image samples based on user requested queries. For example, a user requests a data subset or simply "**bins''**, based on attributes "r" and "petroRad_r". The user can select any of the data subset under this module to create the next data subset.
- Module (4) "**Operator selection**" is a drop-down menu that allows the user to select and execute commands from a data subsetting operator (see Deliverable D3.2 for details about the operators). A set of attributes is presented to the user depending on the operator the user selects.
- Module (5) "**Select the dimensions to group on**" contains a list of attributes required for GDE.
- Module (6) "**Execute**" and "**Undo**" represents the execution of currently selected predicates and undoing of the current pipeline, respectively.
- Module (7) "**Attempts Remaining**" signifies the remaining number of times a user can click on the "Execute" button.
- Module (8) "**End Session**" button allows a user to stop the current phase and redirects the user to a questionnaire.

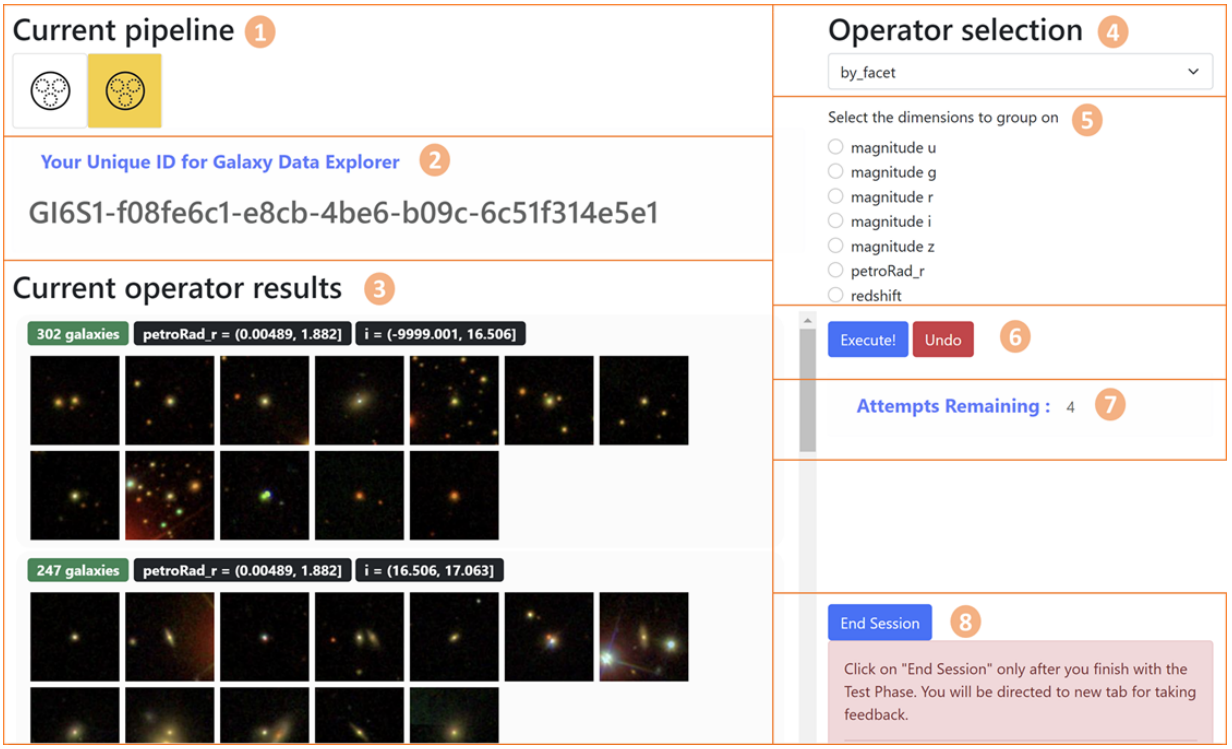


Figure 6.2: *Frontend of the Galaxy Data Exploration toolkit used for qualitative evaluation.*

In this study, the following research question is being investigated:

"**What are the effects of limiting our operators and user interactions on the overall user perception of our GDE system?**"

We develop the two hypotheses to test:

- (H1) Less number of interactions will affect the user's perception of our GDE toolkit positively.
- (H2) More number of operators will affect the user's perception of our GDE toolkit positively.

To test these hypotheses, we devise a *qualitative evaluation approach*. For qualitative analysis, we utilize a 2 x 2 factorial design technique[81] to evaluate the interplay between operators and user interactions and their effect on the user's overall perception of the system. A 2 x 2 factorial design technique/process involves creating a situation, where the participants are exposed to "2" different levels of "2" variables under investigation (and other variables are kept constant). "*This study allows us to deduce the operators that are helpful for users during data exploration*".

[81] K. Haerling Adamson,S. Prion (2020) Two-by-Two Factorial Design, *Clinical simulation in nursing*, *49*, 90–91.

6.4 Experiment Design for Human Factors Qualitative Analysis

A "2 x 2 between subjects' experiment" was performed with the variables as data exploration operators and number of user interactions. The reason for using "between subjects" is to avoid the "learning effect" of the data exploration tool on the user. By varying the factor levels for each independent variable, four levels were obtained that were changed between subjects:

- all-operators vs MIN interactions,
- all-operator vs MAX interactions,
- traditional operators vs MIN interactions,
- traditional operators vs MAX interactions.

The notation "**all-operators**" means the operators *by-facet*, *by-superset*, *by-distribution*, *by-superset* (described in Deliverable D3.2) are at the user's disposal, whereas "**traditional operator**" means only operators *by-facet* and *by-superset* are at user's disposal.

The notation "**MIN**" represents the total length of an expert-created SQL query, whereas "**MAX**" represents an upper bound on the length of the manual pipeline. We choose the upper bound "**MAX**" to be twice the total length of an expert-created SQL query. For example, consider the following expert-created SQL query for finding a particular galaxy data subset.

```
SELECT s.specobjid, s.ra, s.dec
FROM PhotoObj AS p
JOIN SpecObj AS s ON s.bestobjid = p.objid
WHERE p.r BETWEEN 16.928 AND 17.496
        AND p.petrorad_r > 0.00489
```

This expert-created SQL query requires 5 steps: (1) selecting variables, (2) requesting data, (3) joining on certain values, (4) selecting values between certain ranges for variable 1 and (5) selecting values between certain ranges for variable 2. Therefore, in this case "MIN" is 5 and "MAX" equals 10, i.e. twice the size of MIN.

As mentioned before, the expert used for developing the exploration task is from Max Planck Institute and well-versed in SQL querying for SDSS dataset. Now, the notation "MIN interactions" means only the "MIN" number of times a user is allowed to click the "Execute" button of the GDE toolkit (see Figure 6.2).

It should be noted that we use the 'number of SQL predicates used' as a criteria for the number of interactions, because we are comparing between the SQL ease of use and our operators' ease of use. So, if our operators are able to help the users to find the required

dataset within only "MIN" interactions (same as SQL query), then we can conclude that our operators to be as useful as SQL queries in the data exploration process[82].

6.5 Design of Use Case

A data exploration task was designed by an astrophysicist from the Max-Planck-Institut well-versed with the SDSS data exploration task. "*The goal for the user was to utilize the given number of operators and interactions to explore and eventually find a homogeneous subset of* non-dispersed *spiral galaxies*". An example is shown in Figure 6.3. We provide initial training to our users which consists of finding a homogeneous subset of yellow pointed galaxies as shown in Figure 6.4. Both the training and task use cases are commonly-observed examples in the astrophysics community, where an astrophysicist initiates various SQL queries to find a data subset for a given homogeneous subset of galaxies.

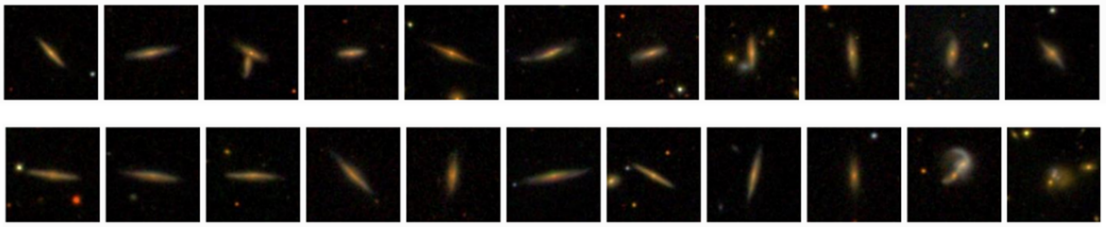


Figure 6.3: *Sample of homogeneous data set showing compact galaxies used during test session*.

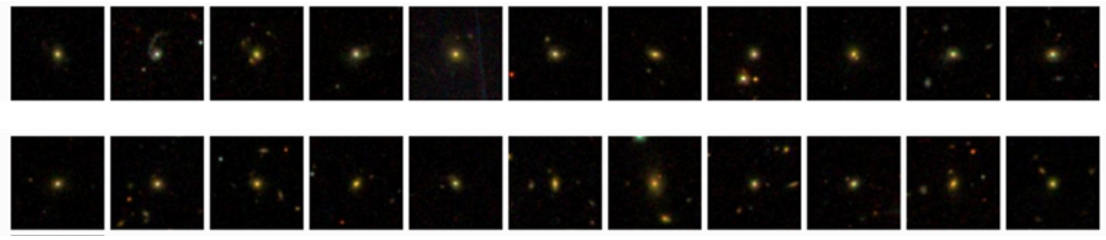


Figure 6.4: *Sample of homogeneous data set showing compact galaxies used for training of users*.

We first ask our expert to complete the SQL query for the data exploration task. The length of the SQL query was 6 and was used as "MIN" for the interactions factor. By using random draw, each user is assigned to a specific number of operators and interactions and was asked to perform the data exploration task. Recruited subjects were redirected to "Google Forms" to complete the consent form and finish with the feedback (Figure 6.6).

[82] Another point should be noted is that the operators can be considered to be more robust than SQL style querying platforms, since one can make synthetic and semantic mistakes when writing SQL queries. However, when using our operators such a case is not observed.

During a data exploration task, the user can explore the data set by creating a predicate. This includes choosing an operator, an attribute and the data subset. Each selection of predicates was considered and recorded as one interaction. By changing the operator, the data subset process was affected and also the time taken to find the required dataset. Also, by varying the number of interactions allowed, the data subset process is affected and results in users requiring more interactions than allowed during the experiment. Due to a varying number of operators and interactions (independent factors), factors (dependent variable) such as Feeling of Accomplishment, Effort required for system use, Mental demand, Perceived Controllability, Temporal demand are also affected.

6.6 Data Collection

For the purpose of data collection, 20 participants were recruited. Figure 6.5 shows their demographics distribution. The participants were randomly assigned to four different groups (as shown in Table 6.1), each group representing a particular set of operators to use and a limited number of interactions the user can perform.

Table 6.1: *Different groups for understanding the effect of change in number of operators & interactions*.

Group	Condition
Group # 1	All-operators vs MIN interactions
Group # 2	All-operators vs MAX interactions
Group # 3	Traditional operators vs MIN interactions
Group # 4	Traditional operators vs MAX interactions

After the participant reads and signs the informed consent form, the training session begins. The aim of the training session is to make the user get used to the exploration tool and task at hand. During the training session, the user performs the data exploration task while getting accustomed to the user interface. After the training session, the experiment or test session begins. During the test session, a unique user ID is assigned to the user by using an automated Universal Unique Identification (UUID) generator for the test webpage. After the user completes the trial, the user ends the trail by clicking on the “End Session”. It redirects the user to a Google form that allows recording the experience concerning perceived difficulty of the task, frustration, ease of finding the dataset, feeling of being restricted, ease of concentration, and user comments.

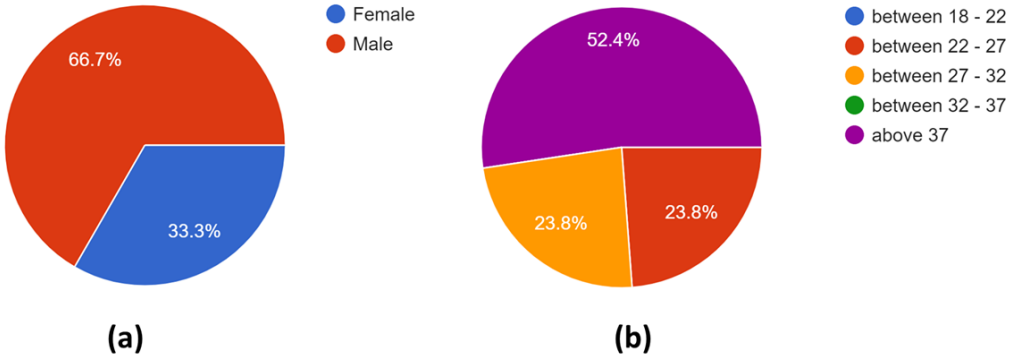
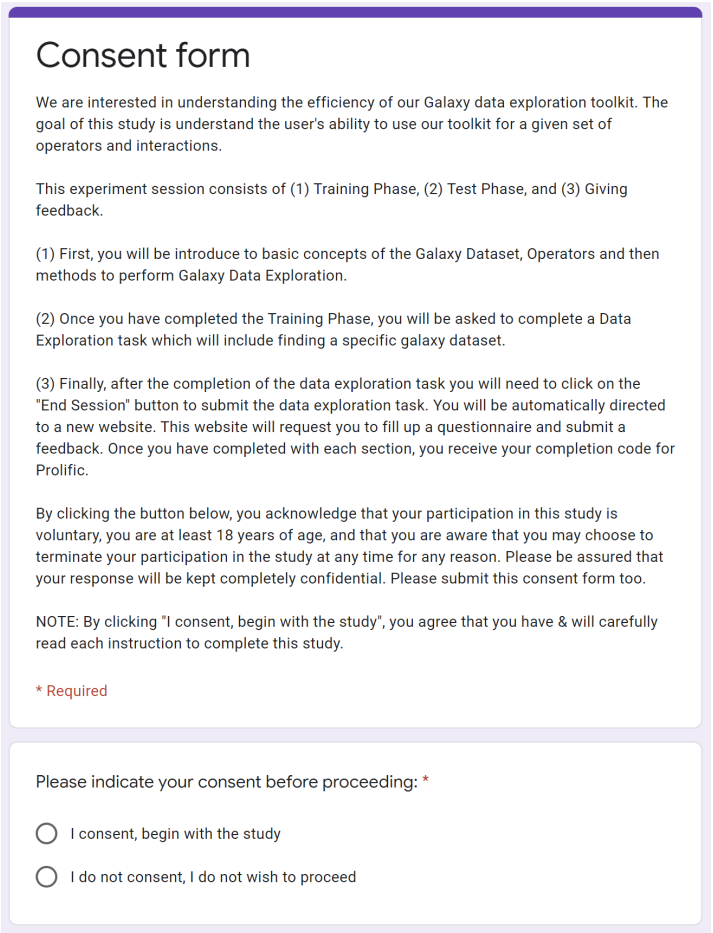


Figure 6.5: *Participants demographics: (a) based on gender and (b) based on age group.*



Consent form

We are interested in understanding the efficiency of our Galaxy data exploration toolkit. The goal of this study is understand the user's ability to use our toolkit for a given set of operators and interactions.

This experiment session consists of (1) Training Phase, (2) Test Phase, and (3) Giving feedback.

(1) First, you will be introduce to basic concepts of the Galaxy Dataset, Operators and then methods to perform Galaxy Data Exploration.

(2) Once you have completed the Training Phase, you will be asked to complete a Data Exploration task which will include finding a specific galaxy dataset.

(3) Finally, after the completion of the data exploration task you will need to click on the "End Session" button to submit the data exploration task. You will be automatically directed to a new website. This website will request you to fill up a questionnaire and submit a feedback. Once you have completed with each section, you receive your completion code for Prolific.

By clicking the button below, you acknowledge that your participation in this study is voluntary, you are at least 18 years of age, and that you are aware that you may choose to terminate your participation in the study at any time for any reason. Please be assured that your response will be kept completely confidential. Please submit this consent form too.

NOTE: By clicking "I consent, begin with the study", you agree that you have & will carefully read each instruction to complete this study.

* Required

Please indicate your consent before proceeding: *

I consent, begin with the study

I do not consent, I do not wish to proceed

Figure 6.6: *Screenshot of consent Form for user studies.*

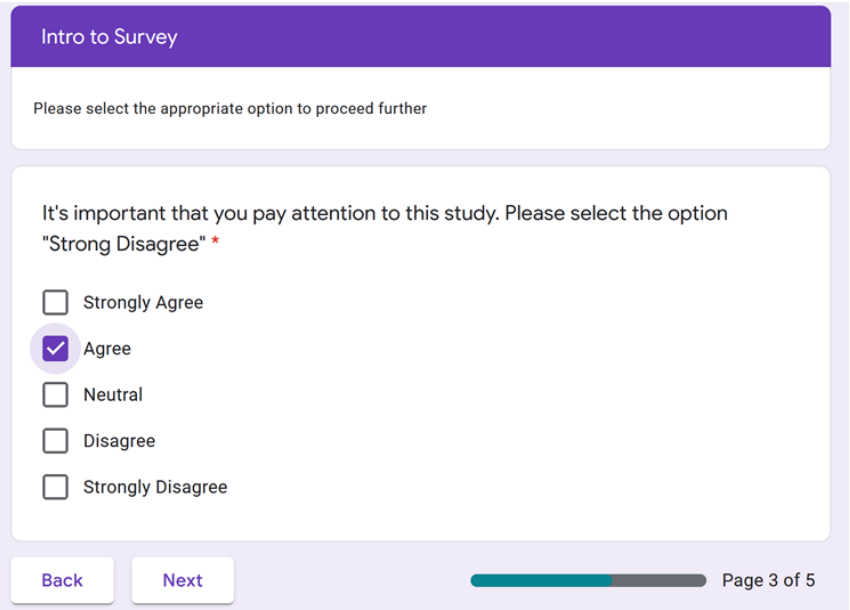


Figure 6.7: *Screenshot of attention test for user studies.*

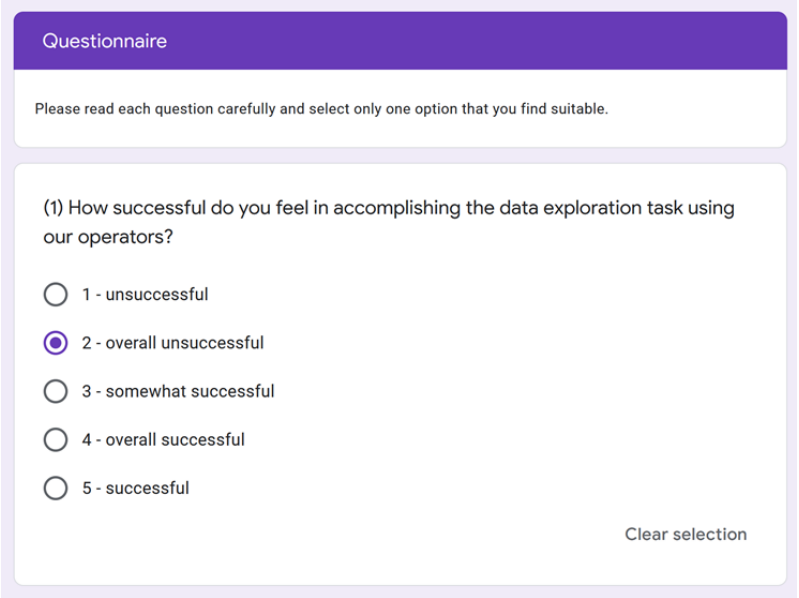


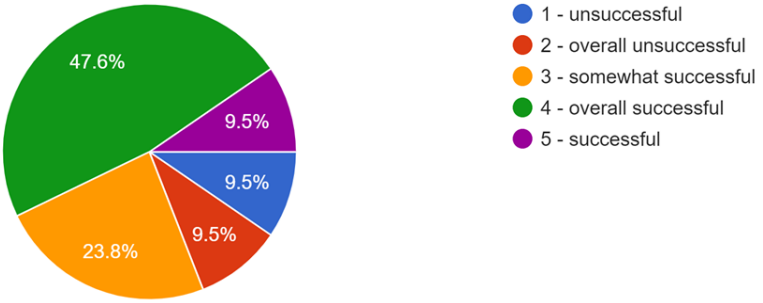
Figure 6.8: *Screenshot for a sample question.*

6.7 Results of User Feedback

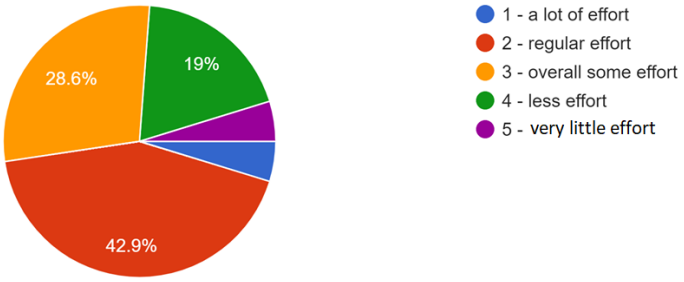
At the end of each trial, each user was requested to fill out a questionnaire regarding the data exploration experience. The questionnaire was designed as a modified version of NASA

TLX[83] for system interaction experience and consisted of 4 questions regarding Feel of Accomplishment (**Question 1**), Effort required during System Usage (**Question 2**), Mental Demand (**Question 3**), and Perceived Controllability (**Question 4**). The users were asked to give a score on a Likert scale of 5 to 1 (5: very favorable, 1: unfavorable). Results for each question for all participants are summarized below by using pie charts.

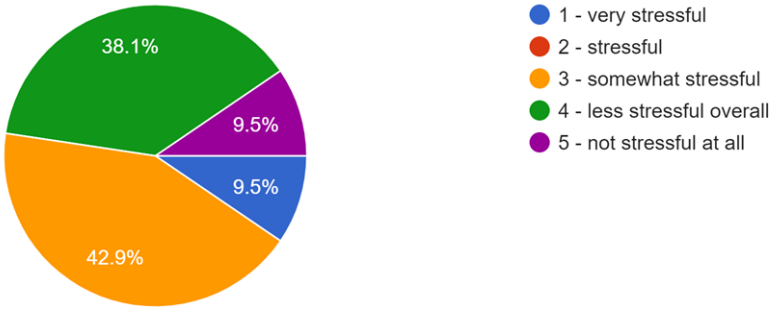
Question 1: How successful do you feel in accomplishing the data exploration task in the test phase using the given set of operators?



Question 2: How much effort do you think was required in the test phase to search the target data subset using the given set of operators?

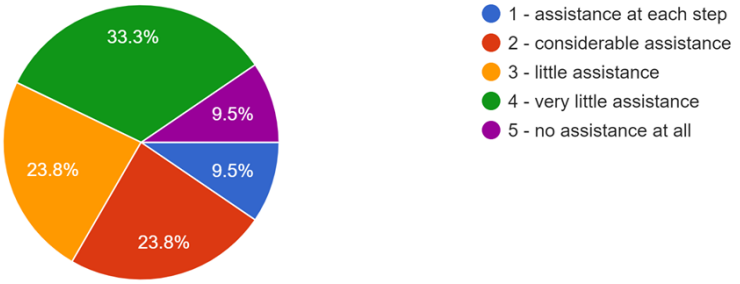


Question 3: How mentally demanding was the Test Phase using the given set of operators?



[83] Q. Roy, F. Zhang, D. Vogel (2019), Automation Accuracy Is Good, but High Controllability May Be Better, CHI Conference.

Question 4: How much assistance do you think you would have required during the test phase, if we were to provide you with an expert?



Results from the questionnaire for different levels of operator and interactions (or simply Groups) are summarized in Figure 6.9.

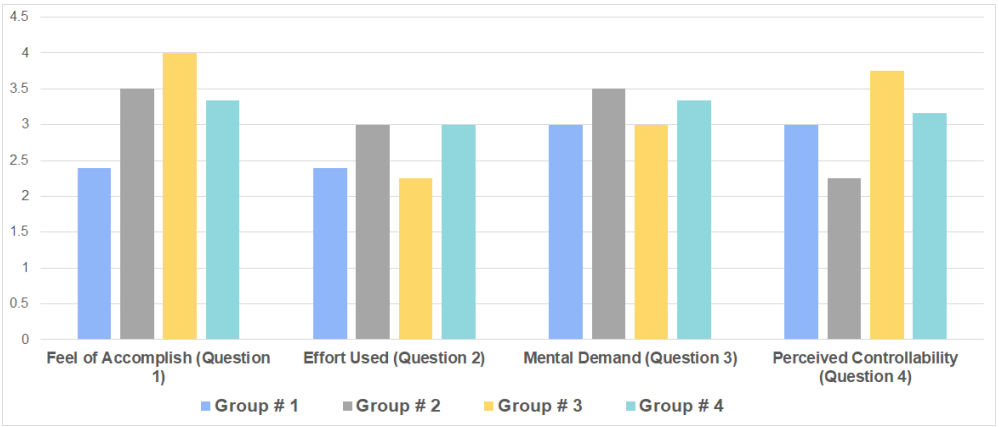


Figure 6.9: *Results for the questionnaire for different levels of operator and interactions.*

Table 6.2: *Result for Kruskal-Wallis Significance test for four questions.*

Questions	H-stat value	p-value
Question 1	8.3758	0.03885*
Question 2	2.1414	0.5436
Question 3	1.0664	0.7852
Question 4	0.7823	1.0783

6.8 Summary

The results of the NASA-TLX scale for our subjective questionnaire are shown in Figure 6.9 and Table 6.2. A Kruskal-Wallis[84] significance test was performed for each variable (e.g. Feel of Accomplishment, Effort used, etc.) to see if there is a difference in the mean values for the four groups shown in Table 6.1.

Only variable "Feeling Accomplished" showed significant effect due to varying number of operators and number of allowed interactions. In case of variable "Feeling of Accomplishment", Group # 3 (Traditional operators vs MIN interactions) showed the most favorable condition by the participants. This reveals that traditional operators and min interactions favors the user's ability of feeling accomplished, thus supporting hypothesis H1 only.

A pairwise Kruskal-Wallis test showed a significant difference between Group # 1 and Group # 3, indicating that users favored Group # 3 more than Group # 1. Other groups showed no pairwise differences, indicating that no significant effect on users' perception of accomplishment of the task for a given set of operators and interactions.

Now that we have evaluated the pipeline component qualitatively, our next aim is to focus on quantitatively evaluating the data exploration process carried out by each user during the test phase in this study. This will include parsing each user data from logs and then applying statistical analysis to understand which group (mentioned in Table 6.1) was more effective in using our GDE toolkit. The main research question under investigation is "What factors limit the user's ability to reach their data exploration goal?" We will do that by crafting appropriate statistical analysis methods.

The feedback questionnaire included open-ended question about what the users **liked** the most and the **least** about the trial, and if they had any additional comments or suggestions. There were various comments related to the experimental setup in general. User # 4 reported "*I think that the guidelines were sufficient*.", User # 14 reported "*The guideline is well written and easy to follow*", and User # 15 reported "*The guidelines in the presentation were good*". While some of the other comments were: User # 1 "*Please state the objective of the study upfront. It was not clear*", User # 17 "*It takes time to read and understand the operators and how they are working*." ,User # 18 "*Even though I glanced through the training and I know the operators, I was not really sure which task I was supposed to perform. There should be clearer instructions*.", User # 20 "*I didn't understand what to do in the first place, during the real test the 2 things we aim for are not clear enough*".

One of the commonly observed comments was related to the theory and understanding of terms related to astronomy. For example, User # 2 noted "*I found it quite easy. My main problem was that I am not familiar with the astrophysics variables and concepts, so I was not*

[84] Kruskal, W.H., Wallis, W.A. (1952), Use of ranks in one-criterion variance analysis. J. Am. Stat. Assoc. 47, 583–621 and errata, ibid. 48, 907–911

sure that my result is correct", while User # 5 noted "*I needed a lot of time in the training phase as I was not familiar with the astronomy terminology. Overall it was quite difficult for me, however, I think that the guidelines were sufficient*", while User # 10 summarized "*Being no astronomer, the images itself were rather useless for me - i had to look at the numbers over and over again. Also, being no native speaker, to me, "relatively far" and "far away" are both very far away, and my task was to find galaxies "near by", which was not defined*".

A good solution to these problems stated will be to clearly state the goal of the study at the beginning of the study (e.g. on Consent form) and also provide a video presentation (along with powerpoint presentation) on the theory of the study subject and on the flow of the experiment. The presentation should be re-visited to avoid any vague terms and superficial explanations for participants from non-astronomical backgrounds.