Practical Linked Data Access via SPARQL: The Case of Wikidata

Adrian Bielefeldt
cfaed, TU Dresden
Dresden, Germany
adrian.bielefeldt@tu-dresden.de

Julius Gonsior
cfaed, TU Dresden
Dresden, Germany
julius.gonsior@tu-dresden.de

Markus Krötzsch
cfaed, TU Dresden
Dresden, Germany
markus.kroetzsch@tu-dresden.de

ABSTRACT

SPARQL is one of the main APIs for accessing linked data collections. Compared to other modes of access, SPARQL queries carry much more information on the precise information need of users, and their analysis can therefore yield valuable insights into the practical usage of linked data sets. In this paper, we focus on Wikidata, the knowledge-graph sister of Wikipedia, which offers linked data exports and a heavily used SPARQL endpoint since 2015. Our detailed analysis of Wikidata's server-side query logs reveals several important differences to previously studied uses of SPARQL over large knowledge graphs. Wikidata queries tend to be much more complex and varied than queries observed elsewhere. Our analysis is founded on a simple but effective separation of *robotic* from *organic* traffic. Whereas the robotic part is highly volatile and seems unpredictable even on larger time scales, the much smaller organic part shows clear trends in individual human usage. We analyse query features, structure, and content to gather further evidence that our approach is essential for obtaining meaningful results here.

1 INTRODUCTION

The SPARQL query language [7] is one of the most powerful and most widely used APIs for accessing linked data collections on the Web. Large-scale RDF publication efforts, such as DBpedia [2], routinely provide a SPARQL service, often with live data. Moreover, SPARQL has been an incentive for open data projects that are based on other formats to convert their data to RDF in order to improve query functionality, a route that was chosen by large-scale projects such as Bio2RDF [1] or the British Museum.[1] One of the most prominent such project is *Wikidata* [17], the large[2] knowledge graph of Wikipedia, which is offering browsable linked data, RDF exports, and a live SPARQL service since September 2015.

Analysing the queries sent to SPARQL services promises unique insights into the practical usage of the underlying resources [12]. This opens the door to understanding computational demands [3, 9, 13], improving reliability and performance [10], and studying user behaviour [14, 15]. Data providers in addition are highly interested in learning how their content is used. This research is enabled by more and more datasets of SPARQL query logs becoming available [11, 16]. In some cases, including Wikidata, access to SPARQL logs remains strongly regulated, but research can help to safely publish useful data there as well (as we intend as part of our work).

Unfortunately, the initial enthusiasm in the study of practical SPARQL usage has been dampened by some severe difficulties. A well-known problem is that SPARQL services experience extremely heterogeneous traffic due to their widespread use by software tools [14, 15]. Indeed, a single user's script can dominate query traffic for several hours, days, or weeks – and then vanish and never run again. Hopes that the impact of such extreme events would even out with wider usage have not been justified so far. Even when studying the history of a single dataset at larger time scales, we can often see no clear usage trends at all. For example, Bonifati et al. recently found that within the years 2012–2016 the keyword DISTINCT was used in 18%, 8%, 11%, 38%, and 8% of DBpedia queries, respectively [3].

As a result, the insights gathered by SPARQL log analysis so far have remained behind expectations. It seems almost impossible to generalise statistical findings, or to make any predictions for the next year (or even month). Building new optimisation methods or user interfaces based on such volatile findings seems hardly worth the effort. And yet, even recent research works rarely make any attempt to quantify or at least discuss the impact of (random) scripts on their findings. Exceptions are few: Raghuveer hypothesised that similarities in query patterns can be used to find bots, and provided basic analysis of bot requests isolated from larger logs [14]; Rietveld and Hoekstra used client-side SPARQL logs as a subset of true user queries that they compared to server-side logs [15].

In this work, we take a first look at the SPARQL usage logs of the official Wikidata query service, and we ask if and how relevant insights can be obtained from them. Our starting hypothesis is that SPARQL queries can be meaningfully partitioned into two classes: *organic queries* fetch data to satisfy an immediate information need of a human user, while *robotic queries* fetch data in an unsupervised fashion for further automated processing. We then classify queries accordingly based on user agent information, temporal distribution, and query patterns, and conduct further analysis on the results. We argue that the organic component of SPARQL query traffic can then be studied statistically, since it is relatively regular and since it can reveal the needs of many actual users. In contrast, the robotic component of query traffic should rather be subjected to a *causal* analysis that attempts to understand the sources of the traffic, so as to predict its current and future relevance to answering specific research questions.

Our main contributions are as follows:

(1) We propose the concept of organic and robotic SPARQL traffic as a basic principle for query log analysis.
(2) We present a method for classifying query logs accordingly, and we use it to partition a set of over 200M Wikidata

[1] http://www.britishmuseum.org/about_us/news_and_press/press_releases/2011/semantic_web_endpoint.aspx (accessed January 2018)
[2] >45M entities, >400M statements, >200K editors (>37K in Jan 2018), >640M edits

LDOW'2018, April 2018, Lyon, France

SPARQL queries. Only 0.31% of the queries are organic, supporting our thesis that human information need is completely hidden by bots in most published analyses.

(3) We evaluate our classification by analysing several aspects that we consider characteristic for organic and robotic queries, respectively. This supports our conjecture that we can effectively distinguish the two types of traffic.
(4) We investigate the organic Wikidata traffic to gain basic insights into actual direct usage of Wikidata.
(5) We discuss anonymisation aspects and potential privacy issues, which forms the basis for our ongoing efforts for allowing the Wikimedia Foundation to release essential parts of our datasets to the public.

Besides the concrete contributions towards understanding the use of Wikidata, we believe that our systematic study can help in advancing the research methodology in the wider field of analysing linked data access through rich query APIs. Indeed, due to the versatile use of SPARQL services – for manual and for automated requests, for transactional and analytical queries, interactively or in batch processes – the analysis of their usage requires suitable techniques and methods that are not sufficiently developed yet.

2 SPARQL ON WIKIDATA

Wikidata is the community-created knowledge base of the Wikimedia Foundation. It was founded in 2012 with the main goal of providing a central place for collecting factual data used in Wikipedia across all languages [17]. As of March 2018, Wikidata stores more than 402 million statements about over 45 million entities.[3] The data is collaboratively curated by a global community, with over 18,000 registered editors making contributions each month. Wikidata is widely used in diverse applications, such as Apple's Siri mobile app, Eurowing's in-flight information system, and data integration initiatives such as the Virtual Integrated Authority File.

The data model of Wikidata is based on a directed, labelled graph where *entities* are connected by edges that are labelled by *properties*. Entities can have labels in many languages, but their actual identifiers are abstract: properties use identifiers such as P569 ("date of birth"), while other entities (called "items") use identifiers such as Q42 ("Douglas Adams"). Both types of entities can be freely created by users. The model is distinct from RDF in that edges of the graph may in turn have annotations. This feature is used to record sources, temporal validity, or other contextual information. Indeed, annotations on edges are using the same community-defined vocabulary as edge labels.

Since September 2015, Wikidata provides an official SPARQL service (user interface at https://query.wikidata.org/) to query its data. For this purpose, data is first converted to RDF. Each edge is represented by a URI that can be associated with its annotations, following an encoding as laid out by Erxleben et al. [5]. In addition to this faithful encoding, the RDF export also includes *simplified* statements that only capture the actual edge as a single RDF triple, without any of its annotations. Different URIs are introduced for the different roles that properties can play in this encoding, so that all views of the data can be stored and queried in one database without risk of confusion. As of March 2018, the RDF encoding contains about 4.7 billion triples.[4] Queryable data is updated at least once per minute to keep synchronised with updates.

Table 1: Countries by number of cities with a female mayor

```
SELECT ?country ?countryLabel (count(*) AS ?count)
WHERE {
    ?mayor wdt:P21 wd:Q6581072 .
    ?city wdt:P31/wdt:P279* wd:Q515 .
    ?city wdt:P17 ?country .
    ?city p:P6 ?statement . ?statement ps:P6 ?mayor .
    FILTER NOT EXISTS { ?statement pq:P582 ?x }
    SERVICE wikibase:label {
        bd:serviceParam wikibase:language "ru,en" .
    }
}
GROUP BY ?country ?countryLabel
ORDER BY DESC(?count) LIMIT 100
```

The Wikidata SPARQL service is based on the BlazeGraph database management system. SPARQL support is mostly standard, but includes some built-in operational extensions that are represented by (ab)using SPARQL's SERVICE directive, which is normally used for federated queries to external services. Of chief practical importance is the *labelling service*, used to fetch optional entity labels with support for fallback languages. The widespread use of this service does affect the structure of queries, which rarely include the otherwise familiar OPTIONAL-FILTER pattern to select labels in the desired language.

Table 1 shows an example query that illustrates several aspects of Wikidata's RDF encoding. The query returns the 100 countries that have the most cities with a female mayor. Within the query pattern, the first line finds a value for `?mayor` with *gender* (P21) *female* (Q6581072). We are using the simplified property `wdt:P21`, which cannot have annotations or source information to its triples. The following line finds a `?city` that is *instance of* (P31) of the class *city* (Q515), or of any *subclass* thereof (P279*). We then determine *country* (P17) of this city. The fourth line of the pattern then uses the more complex RDF encoding to find a `?statement` for property *mayor* (P6) and matches its value to `?mayor`. We require that this statement has no *end time* (P582) to ensure that the mayor is current. Finally, the service `wikibase:label` is invoked to fetch labels in Russian, or, as an alternative, English. The query can readily be executed online.

Wikidata provides extensive documentation for using SPARQL,[5] including a collection of over 300 example queries and pages to request help in writing new queries. The query service receives several million queries per day.

3 WIKIDATA SPARQL QUERY LOGS

We now give an overview of the datasets we are working with for this paper. Our data is based on the server-side request logs (Apache Access Log Files) of the Wikidata SPARQL query service, as exported from the internal logging infrastructure of Wikimedia. As logs contain sensitive information (especially IP addresses), this

[3] see https://www.wikidata.org/wiki/Wikidata:Statistics and links therefrom

[4] https://grafana.wikimedia.org/dashboard/db/wikidata-query-service

[5] https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/queries/

Table 2: Query Dataset Sizes

	Total	Valid	Unique	Patterns
I1	70,201,736	61,250,218	12,833,923	502,083
I2	73,425,065	71,853,238	19,229,539	216,126
I3	79,797,901	78,600,433	26,442,258	180,605

Table 3: Example pattern for the query in Table 1

```
SELECT ?var1 ?var2 (count(*) AS ?var3)
WHERE {
    ?var4 wdt:P21 wd:QName1.
    ?var5 wdt:P31/wdt:P279* wd:QName2 .
    ?var5 wdt:P17 ?var1 .
    ?var5 p:P6 ?var6 . ?var6 ps:P6 ?var4 .
    FILTER NOT EXISTS { ?statement pq:P582 ?x }
    SERVICE wikibase:label {
        bd:QName3 wikibase:language "string1" .
    }
}
GROUP BY ?var1 ?var2
ORDER BY DESC(?var3) LIMIT 1
```

data is not publically available, and all records are deleted after a period of three months. For this research, we therefore created less sensitive (but still internal) snapshots that contain only SPARQL queries, request times, and user agent information, but no IPs.

We consider the complete query traffic in three consecutive intervals in 2017, each spanning exactly four weeks: (I1) 12th June–9th July, (I2) 10th July–6th August, and (I3) 7th August–3rd September. We process all queries with the Java module of Wikidata's BlazeGraph instance,[6] which is based on OpenRDF Sesame, with minimal modifications in the parsing process re-implemented to match those in BlazeGraph. In particular, BIND clauses are moved to the start of their respective sub-query after the first parsing stage.

This results in 211,703,889 valid queries. We then eliminate exact string duplicates individually for each interval to obtain a subset of unique valid queries. A specific unique query may therefore still re-occur in several intervals. The numbers of queries per interval are given in Table 2.

The column *Patterns* counts unique query patterns after a further abstraction. We uniformly replace all resources in subject and object positions with a normalised placeholder, using different placeholders for URIs and literals (by type). Patterns therefore reflect basic resource types, co-occurrence of resources, and all predicates used in the query. We also normalise values in LIMIT and OFFSET. Table 3 illustrates the pattern we would obtain for the query from Table 1. Our approach follows Raghuveer who observed that programs often use query templates to construct many similarly looking queries where only certain placeholders are instantiated [14]. However, we do retain predicates since the abstraction would otherwise be too strong (in particular, most one-triple queries would lead to the same pattern when also abstracting predicates).

We can see a clear trend towards a significant increase in query traffic over time, which we have witnessed for many months since the introduction of the query service, the exception being a decline in June 2017 following new measures for throttling scripts that send overly many queries in very short times. We can also see some variability in the number of query types, which roughly measures how uniform queries were in an interval.

Table 4: An idealised view of organic and robotic queries

	Organic queries . . .	**Robotic queries . . .**
a . . .	fetch data to be delivered directly to human users	. . . fetch data to be processed algorithmically
b . . .	are part of an ongoing human interaction	. . . are executed without close human supervision
c . . .	reflect an immediate human information need	. . . may serve many indirect purposes (or none)
d . . .	are typically sent from browser applications	. . . are sent from applications that rarely run in browsers
e . . .	represent the needs and interests of many	. . . are not representative of general needs
f . . .	are relatively diverse	. . . are relatively uniform
g . . .	have uniform distributions that change continuously	. . . have skewed distributions that change abruptly

4 CLASSIFYING QUERY SOURCES

We conjecture that a meaningful analysis of SPARQL query logs in most cases must involve a classification of traffic into two basic forms, *organic* and *robotic* queries. In this section, we characterise both types, and we present our approach for separating them for the example case of the Wikidata query logs.

In an idealised view, organic and robotic queries are characterised as shown in Table 4. Note that we do not restrict organic queries to mean those that are manually typed in by individual users, as studied previously [15]. Indeed, we wish to include users who are not aware that SPARQL is being used at all, as long as the query is representative of their present information needs.

Nevertheless, there is a grey area of applications that may allow users to schedule and execute thousands of queries through a browser interface. There is a gradual transition between idealised organic and robotic queries, and, in theory, many intermediate applications are conceivable. If in doubt, such cases should be considered robotic, since they can then still be analysed individually without their traffic giving undue prominence to individual users among the organic queries.

To classify queries as organic or robotic, we rely on just two characteristics, (d) and (f), with some further guidance from (g). We use only three features to decide the type of a query: the user agent (for (d)), the abstract query pattern (for (f)), and (when available) comments in the query that some tools use to identify themselves. Our implementation determines the query type by custom rules that use these two aspects only. By default, we expect all browser-related user agents to indicate organic traffic, while all other agents indicate robotic traffic. However, we have implemented a more fine-grained causal analysis that tries to relate certain user-agent/pattern combinations to specific sources (programs). Individual sources can then be manually added to either of the two categories.

[6] Maven artefact com.blazegraph.sparql-grammar v2.1.4

Table 5: Robotic and Organic Dataset Sizes

		Valid	Unique	Un./Val.	Patterns
robotic	I1	61,052,879	12,690,953	20.79%	461,974
	I2	71,648,854	19,080,323	26.63%	175,432
	I3	78,343,266	26,239,839	33.49%	145,286
organic	I1	197,339	142,970	72.45%	40,109
	I2	204,384	149,216	73.01%	40,694
	I3	257,167	202,419	78.71%	35,319

Concretely, we consider *WikiShootMe* (nearby sites to photograph), *SQID* (data browser), and *Histropedia* (interactive timelines) as sources of organic traffic, and leave other software tools in the robotic category. On the other hand, we had frequent occasion of classifying queries sent from well-known browsers as robotic. Examples included an apparently browser-based application that retrieved Wikidata items with hundreds of thousands of diverse movie database identifiers, and another "browser" that issued the exact same query three million times. Clearly, these cases did not meet our criteria for organic traffic, yet they would completely change the characteristics of the organic dataset when not discovered.

We have applied this classification to the set of valid queries, leading to a distribution of queries as shown in Table 5. The total of 658,890 organic queries accounts for less than 0.5% of the traffic, and would therefore be overlooked completely in any non-discriminative analysis. Already Table 5 clearly shows that this small fraction of traffic behaves significantly different from the overall dataset. While robotic traffic contains 21% to 33% unique queries, organic queries are 72%–79% unique. The non-discriminative query analysis of Bonifati et al. showed that DBpedia logs have between 30% (2016) and 54% (2013) unique queries, while other datasets are 3%–30% unique [3]. Our robotic queries therefore are well within the typical range observed so far, whereas our organic queries seem to represent a very different type of traffic.

4.1 SPARQL Feature Prevalence

Further differences are revealed when analysing the SPARQL query features used in each dataset. SELECT queries make up more than 99% of queries in each dataset, so we do not report uses for DESCRIBE, CONSTRUCT, or ASK. Table 6 shows the prevalence of most common solution set modifiers, graph matching patterns, and aggregates in each of the query sets. *Join* refers to the (often implicit) SPARQL join operator; *Filter* also includes FILTER expressions using NOT EXIST; and SERVICE calls are split between the very common Wikidata label retrieval service (*lang*) and others. We will discuss several aspects of this large table in the remainder of this section.

Absent Features. We have omitted features that were generally used in less than 1% of queries from Table 6. This applies to REDUCED, EXISTS, GRAPH, the occurrence of + in property paths, as well as specific aggregation functions (MIN, MAX, SUM, AVG). Any other feature that is not shown in the table has not been counted. The absence of GRAPH in our data is expected, since Wikidata does not use named graphs.

Unique Queries. Many studies restrict to unique queries. We do not find it obvious that this is most suitable. Indeed, in a continuously changing database like Wikidata, where the RDF export is updated at least once per minute, a repeated query may indicate a real and recurring information need, and it may always require a new answer to be computed. Moreover, considering organic traffic, it is also relevant if many users require the exact same data. Therefore neither human interests nor database query load can necessarily be understood any better by eliminating duplicates. Table 6 includes prevalence among unique queries to show that the choice between the two views has a significant impact on results. A higher prevalence among unique queries also indicates that queries with that feature are more likely to be repeated than queries without this feature. For example, organic queries with subqueries are less frequent when eliminating duplicates, which shows that such queries tend to re-occur more often than others. Note that not all queries contribute to the feature counts in Table 6, and in general it is possible that queries without any counted feature (e.g., single triple queries) are much more common among the unique queries.

Stability of Results. The usage patterns of any technical system will evolve over time, but it is difficult to estimate the stability of specific metrics. We have split our data into three intervals to make such changes visible. Robotic traffic exhibits huge fluctuations, e.g., for joins (67%–88%) and OPTIONAL (11%–25%). Prevalence among unique queries sometimes fluctuates independently, e.g., for UNION (2.5%–8.6%). By studying one or several of the intervals as a single dataset, and by choosing to include or exclude duplicate queries, one could therefore arrive at extremely different conclusions from this data. Many previous studies of SPARQL logs – often smaller than ours, and therefore even more easily dominated by bots – should be viewed in this light.

Organic traffic tends to be more stable, but also shows significant variations in some cases. Especially we can see some change in I3, most notable for VALUES and UNION. We discuss and explain this change in detail in Section 6. In most cases, however, we can see that metrics are fairly continuous, and that moreover, unique queries are much more representative of all queries than in the robotic case.

Feature Usage as Compared to Other Studies. The previous discussion suggests that it is generally questionable whether any insights can be obtained by comparing SPARQL log metrics based on a non-discriminative analysis of all queries. Nevertheless, there are some aspects where our results show overwhelming differences from previously reported findings. The most systematic overview across several datasets is given by Bonifati et al. [3]. They found SERVICE, VALUES, BIND, and property paths to occur in less than 1% of unique queries, while they have great relevance in all parts of our data. Similarly low prevalence was reported for subqueries, SAMPLE, and GroupConcat – our robotic traffic is similar, but our organic traffic paints a very different picture. Especially subqueries are strikingly common there.

4.2 SPARQL Feature Co-Occurrence

The expressive power and computational complexity of SPARQL queries is determined not so much by the presence or absence

Table 6: Relative prevalence of SPARQL features among valid queries (among unique queries in parentheses)

	organic			robotic		
Feature	I1	I2	I3	I1	I2	I3
Limit	31.08% (31.04%)	39.55% (38.69%)	46.56% (48.13%)	21.12% (26.45%)	16.86% (13.23%)	17.42% (19.55%)
Distinct	26.50% (23.12%)	31.40% (26.06%)	19.05% (15.66%)	15.84% (30.45%)	5.48% (10.18%)	4.27% (6.85%)
Order By	17.29% (16.66%)	14.75% (14.23%)	13.22% (10.40%)	12.97% (8.06%)	8.01% (6.49%)	6.78% (1.19%)
Offset	0.40% (0.51%)	2.92% (3.51%)	0.37% (0.42%)	7.73% (15.81%)	6.07% (3.62%)	6.29% (13.17%)
Join	87.59% (85.41%)	87.82% (85.60%)	89.76% (89.38%)	88.48% (71.76%)	78.53% (67.11%)	67.41% (54.89%)
Optional	42.36% (44.74%)	46.24% (46.36%)	55.92% (63.14%)	25.08% (34.61%)	11.63% (12.93%)	11.45% (9.41%)
Filter	25.89% (23.49%)	29.12% (27.61%)	22.24% (17.27%)	21.64% (23.40%)	17.92% (13.52%)	13.79% (13.65%)
Path with *	15.02% (13.55%)	15.59% (16.18%)	12.88% (12.41%)	16.43% (9.30%)	19.19% (17.34%)	14.80% (7.66%)
Subquery	13.09% (9.07%)	15.30% (8.77%)	12.79% (7.61%)	0.34% (1.45%)	0.28% (0.82%)	0.33% (0.47%)
Bind	9.85% (9.03%)	9.23% (8.60%)	8.68% (7.11%)	16.29% (13.08%)	12.07% (12.60%)	9.60% (4.51%)
Union	5.10% (3.66%)	5.76% (5.06%)	12.62% (14.40%)	11.26% (8.62%)	8.63% (8.50%)	7.61% (2.53%)
Values	4.44% (4.29%)	3.07% (3.13%)	10.88% (12.63%)	35.72% (10.68%)	30.74% (8.06%)	28.92% (6.24%)
Not Exists	3.31% (2.75%)	3.37% (3.08%)	2.46% (1.65%)	0.19% (0.12%)	0.21% (0.18%)	0.19% (0.07%)
Minus	2.04% (1.99%)	2.91% (3.13%)	1.60% (1.60%)	0.53% (1.03%)	0.92% (1.52%)	1.07% (1.60%)
Service (lang)	44.63% (42.51%)	42.09% (43.53%)	54.78% (59.59%)	10.40% (23.66%)	6.15% (10.90%)	4.27% (6.35%)
Service (other)	11.49% (15.00%)	10.53% (13.73%)	10.32% (12.45%)	4.51% (2.89%)	0.19% (0.44%)	1.16% (1.48%)
Group By	17.12% (13.74%)	19.93% (14.02%)	13.04% (9.54%)	0.41% (0.57%)	0.37% (0.43%)	0.48% (0.30%)
Sample	8.85% (6.45%)	10.93% (5.79%)	4.60% (3.66%)	0.04% (0.04%)	0.04% (0.05%)	0.06% (0.07%)
Count	7.55% (6.66%)	7.60% (7.64%)	8.15% (5.74%)	1.15% (2.38%)	4.30% (7.19%)	0.30% (0.06%)
GroupConcat	1.80% (1.79%)	2.79% (2.26%)	1.17% (1.05%)	0.06% (0.21%)	0.09% (0.17%)	0.02% (0.04%)
Having	1.17% (1.26%)	1.14% (1.35%)	0.72% (0.71%)	0.01% (0.01%)	0.01% (0.02%)	0.00% (0.00%)

of individual features, but by their co-occurrence and interaction. Investigating which combinations of features occur in queries is therefore of interest. We present the results of this analysis here.

Many studies restrict to analysing co-occurrence of join, OPTIONAL, UNION, and FILTER. In our case, however, we find that also subqueries, property paths, VALUES, and SERVICE occur in a significant share of queries. The use of SERVICE is dominated by Wikidata's labelling service, which is mostly used to add labels after fetching query results. To reduce the number of feature combinations, we therefore ignore the labelling service entirely. We do not count queries that use any other type of service, or any other feature not mentioned explicitly (the most common such feature being BIND). Solution set modifiers (LIMIT etc.) and aggregates (COUNT etc.) are ignored: they can add to the complexity of query answering only when used in subqueries, so this feature can be considered an overestimation of the amount of such complex queries.

The results of this analysis are presented in Table 7 for all operator combinations that accounted for more than 1% of queries in either dataset. Path expressions in this case include all queries where either * or + occurs. The features we selected, possibly in combination with a labelling service, account for around 85% of all queries in either dataset, but the distributions are very different.

Traditional Query Fragments. Plain conjunctive queries (CQ) that contain only joins account for 21.3% (30.9%) of organic (robotic) traffic. The frequently studied conjunctive-filter-pattern queries (CFP), which also consider FILTER, increase these values to 29.1% (35.5%). This is far below the prevalence reported for such simple queries in other datasets, typically above 65% [3, 13]. VALUES can be allowed in CQs without an increase in complexity for a coverage of 21.7% (61.4%). Especially robotic traffic contains this pattern, which is not surprising since VALUES is an efficient way to combine query batches into one.

Table 7: Co-occurrence of SPARQL features by query type (Join, Filter, Optional, Union, Path, Values, Subquery)

J	F	O	U	P	V	S	organic	robotic
			(none)				8.04%	19.67%
J							13.29%	11.26%
	F						1.10%	1.92%
J	F						6.68%	2.61%
J				P			2.98%	13.50%
J	F			P			2.48%	0.39%
J					V		0.39%	30.42%
		O					1.26%	0.11%
J		O					22.32%	1.86%
J		O		P			2.07%	0.35%
J	F	O					2.66%	2.13%
J		O	U				3.49%	0.02%
J		O			V		3.38%	0.11%
J		O		P	V		1.01%	0.16%
J						S	2.76%	0.06%
J		O				S	4.78%	0.00%
J	F					S	3.19%	0.03%
J	F	O				S	1.02%	0.00%
Sum of above combinations							82.89%	84.63%
Other counted combinations							6.05%	2.67%
Features not counted							11.06%	12.70%

Path Queries. The extension of CQ with property path expressions leads to conjunctive regular path queries (CRPQs [4, 6]), which account for 24.4% (44.5%) of organic (robotic) queries. While organic queries contain almost the same high amount of path expressions (Table 6), the simple CRPQ fragment does not suffice to capture as many of them as in the robotic case. A highly prominent query fragment for the robotic case are CRPQs with VALUES, which account for 74.9% of all queries.

OPTIONAL and UNION. OPTIONAL is much more popular in organic queries, where together with join it accounts for 44.9% of the traffic. We attribute this to the fact that user interfaces often try to show as much information as available. On the other hand, robotic queries have much less use for query results that may miss some of the queried values, especially since labels can be fetched with a dedicated service. UNION rarely occurs in otherwise simple queries, although it does occur in significant amounts of queries overall. Our findings again deviate from Bonifati et al. who found UNION alone enough to account for 7.5% of unique queries [3].

Subqueries. Table 7 shows that more than 10% of organic queries include subqueries while otherwise using nothing but joins, FILTER, and OPTIONAL. However, we have ignored solution set modifiers and aggregates in this analysis, since they normally play a role only in post processing. In combination with subqueries, however, such operations may have a big impact on the expressive power of the query language.

5 ROBOTIC VS. ORGANIC: EVALUATION

We have already seen that organic and robotic traffic are significantly different in many respects. This shows that our partitioning of queries is not random, but it does not support the claim that they actually can be characterised as in Table 4. In this section, we therefore investigate whether the datasets also exhibit previously asserted characteristics that have not been used to define them in the first place.

Temporal distribution. We begin by considering the temporal distribution. According to Table 4 (b), we would expect organic queries to be correlated with the time of day, while robotic queries should not show any such relationship. Figure 1 shows the hourly temporal traffic volume (in absolute numbers) and the relative distribution across 24h-intervals. Organic traffic follows a strong daily rhythm, with most activities happening during the European and American day and evening. This strongly supports a direct human involvement.

This also suggests that our dataset contains only few organic queries from Asian users, which in our case simplifies the detection of the expected patterns. If usage were globally uniform, it would be promising to correlate daily usage with hints on geographical context, which can be found in queries, e.g., in the form of coordinates or language information.

Figure 1 also shows some abrupt changes in the robotic traffic, as predicted by Table 4 (g).

Content preference. According to Table 4 (c), user queries are expected to mirror more direct human interests. We investigate this by considering the predicates used in queries. Table 8 shows the top ten most frequently used Wikidata properties in organic and in robotic queries. Robotic queries frequently involve IDs of external databases, whereas organic queries often refer to properties related to locations. Human properties such as *occupation* (and, just outside of the top ten, *date of birth* and *sex or gender*) also rank highly in organic queries. *Commons category* refers to Wikimedia Commons and can be used to obtain related media for some entity. *Wiki Loves Monuments ID*, the only identifier in the organic top ten, relates to the eponymous content creation activity of Wikipedia, which aims as gathering more information on local sites of interest.

Table 8: Most frequent Wikidata properties by query type

Organic	Robotic
instance of (P31)	VIAF ID (P214)
image (P18)	Libr. of Congress ID (P244)
coordinate location (P625)	located in . . . (P131)
Commons category (P373)	ISO 3166-2 code (P300)
subclass of (P279)	instance of (P31)
located in . . . (P131)	image (P18)
heritage designation (P1435)	subclass of (P279)
country (P17)	ISSN (P236)
occupation (P106)	MusicBrainz artist ID (P434)
Wiki Loves Mon. ID (P2186)	PubMed ID (P698)

Diversity vs. Uniformity. We have conjectured that organic queries are more diverse, since they are not controlled by a small number of programs, and since user information needs are generally less uniform. Our datasets support this in more than one sense. We have already seen from Table 6 that organic queries tend to use more diverse SPARQL features, including some that have no significant share of robotic queries. We also found that simple combinations of operators can account for 75% of robotic queries, whereas organic queries exhibit a greater variance. For the study in Section 7, we found 17 feature combinations that make up over 1% of organic queries, while only 8 combinations have such high prevalence among robotic queries (Table 7). Similarly, the usage of RDF properties is more skewed in robotic traffic, where only 37 properties occur in more than 1% of queries. In contrast, 59 different RDF properties occur in more than 1% of organic queries.

Another frequently studied structural aspect of queries is their actual length in terms of the number of triples. To determine this number, we have not counted triples that occur within SERVICE clauses, since these are mostly used to call built-in BlazeGraph functions, where triples represent parameters rather than referring to actual RDF data. Figure 2 shows the results. The results show the usual peak of 1-triple queries in the robotic case, with 55.96% of queries having at most this size, and 96% of queries having at most 7 triples. This matches findings for other datasets, where an average of 56.45% of queries had at most one triple and 91% of queries used 6 or fewer [3]. Our robotic average query size of 2.45 triples also resembles that of DBpedia, reported to be between 2.09 and 3.98 for various samples.

In contrast, the size distribution of organic queries is much wider. Queries with at most one triple only make up 17.30%, and one has to consider queries with up to 11 triples to cover 97% of the data. The average query size is 4.38 in this case. The largest organic query

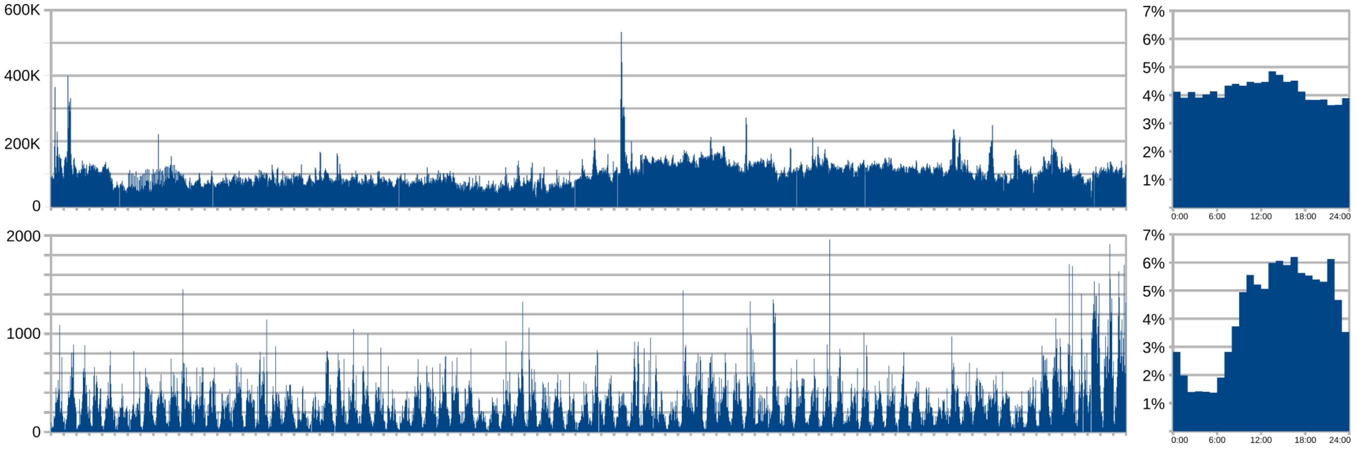

Figure 1: Hourly query volume over 12 weeks (left), and aggregated by time of day, UTC (right); robotic top/organic bottom

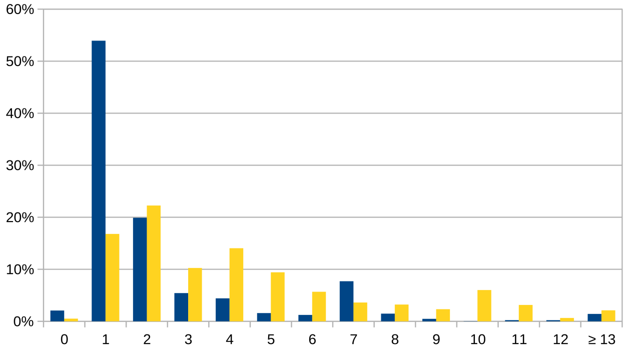

Figure 2: Fraction of queries by number of triples, in each case for robotic (dark, left) and organic (light, right) traffic

had 143 triples, while the largest robotic ones contained 66 triples. Again, we can see that organic queries are more diverse, and also more complex than robotic ones.

6 UNDERSTANDING WIKIDATA USAGE

Based on the above investigations, we assume that our classification of queries can successfully capture the ideas expressed in Table 4. We now turn towards exploiting this insight for obtaining a better understanding of actual Wikidata usage. Much of the previous discussion also contributes towards this goal, e.g., the analysis of temporal distribution (indicating a lack of Asian users) and the ranking of properties (showing significant user interest in local data), but we can refine these findings further.

6.1 Annotations and Complex Statements

As explained in Section 2, Wikidata supports annotations on statements, which allow to express contextual information, and which lead to a more complex RDF encoding. It is therefore relevant to ask if queries make use of such information, or if they rather use the simplified view that drops all annotations.

Table 9: Most frequent Wikidata properties used in queries as annotations on statements (mix of top 5 organic/robotic)

Property	Organic	Robotic
end time (P582)	2.00%	0.74%
start time (P580)	1.97%	0.28%
place of publication (P291)	0.60%	0.00%
Refseq Genome ID (P2249)	0.60%	0.00%
academic degree (P512)	0.47%	0.00%
catalog (P972)	0.04%	0.55%
ticker symbol (P249)	0.04%	0.24%
version type (P548)	0.00%	0.13%

The analysis of URIs used in predicate positions lets us answer this question. References are linked via the RDF provenance vocabulary URI `prov:wasDerivedFrom`, which occurs in 1.1% (0.07%) of all organic (robotic) queries. Another annotation is the statement *rank*, which is used in 0.48% (0.62%) of all queries. Finally, statements can also be annotated with arbitrary Wikidata properties, and such use can also be recognised from URIs. The most frequently used properties in statement annotations for organic and robotic queries are shown in Table 9. Annotations are of course used only in a minority of queries. Their pronounced use in the case of start and end time witnesses the importance of temporal validity when interpreting statements from Wikidata.

Conversely, we can also collect the most common properties used in statements for which annotations are queried. To gauge this metric, we consider RDF properties used for encoding the complex form of statements that uses a dedicated node for representing edges. Note that this does not always indicate that the query also referred to annotations of these properties. Another motivation for using the more complex statement encoding is that the simplified statement encoding is only generated for statements of maximal rank; queries that are interested in all (e.g., historical) data therefore need to use complex statements even when not querying for annotations. Table 10 displays the most common properties that appeared in a form as used for complex statement encodings.

Table 10: Most frequent Wikidata properties of statements whose complex form occurs in queries (mix of top 5 each)

Property	Organic	Robotic
heritage designation (P1435)	7.95%	0.00%
Wiki Loves Mon. ID (P2186)	3.37%	0.00%
coordinate location (P625)	1.32%	0.46%
position held (P39)	1.32%	0.06%
DiseasesDB (P557)	0.63%	0.00%
catalog code (P528)	0.04%	0.55%
PMCID (P932)	0.01%	1.05%
PubMed ID (P698)	0.00%	2.31%
DOI (P356)	0.00%	0.91%

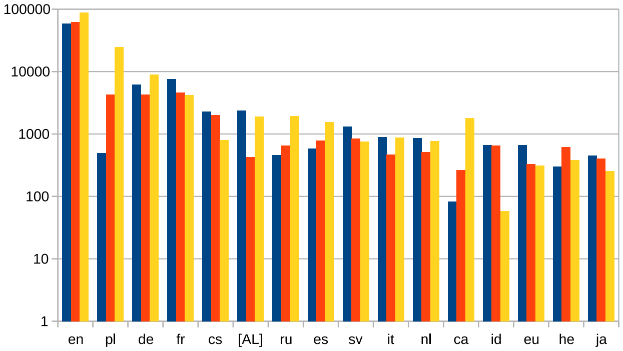

Figure 3: Number of queries for popular primary languages, for each of the three intervals (I1–I3)

We can see that complex statements are a larger fraction in organic queries. Indeed, 18.9% of all organic queries are using properties that are part of the complex RDF encoding of statements, rather than relying on simplified statements represented by single edges alone.

6.2 Language Distribution

We saw that the temporal distribution of organic queries suggests that few users from Asia are accessing Wikidata via SPARQL-based applications. For further insights, it is interesting to study the language-related information contained in queries. Indeed, the use of Wikidata by communities of different languages is a relevant research topic by itself [8].

We have therefore extracted the languages for which queries request labels using the labelling service, and grouped queries by the first (most preferred) language they use. This resulted in over 200 different language tags, including misspellings. Of those, 35 occurred in more than 100 queries. For the 16 languages that occurred in more than 1000 queries, we show the query numbers for each of the intervals in Figure 3 using a logarithmic scale. The special label "[AL]" denotes the value `[AUTO_LANGUAGE]`, used for retrieving a language based on browser settings.

English clearly dominates the list, followed by mostly European languages (cs is Serbian, sv is Swedish, ca is Catalan, and eu is Basque). Asian languages follow only at the end, with Indonesian (id), Hebrew (he), and Japanese (ja) hardly occurring in more than 1,000 queries in total. The sum of several variants of Chinese also amounts to 1,122 queries; lower still is Arabic (798). These observations suggest a strong imbalance in the global use of Wikidata via SPARQL. The choice of language may also reflect the lack of labels in some languages [8], but since the labelling service supports any number of fallback languages, users could still always put their preferred language at the front. Moreover, the amount of queries asking for a certain language is only weakly related to the total number of labels available in a language. As of March 2018, Wikidata has most labels for English (32.7M), Dutch (10.7M), French (9.4M), German (8.0M), Spanish (6.5M), Italian (6.1M), Swedish (6.1M), Russian (5.4M), Cebuano (5.0M), and Bulgarian (2.9M). Polish follows at fourteenth place with 2.6M labels; Hebrew is only at 92nd place with less than 460K labels.

The situation for robotic traffic is similar, but the labelling service is used in a much smaller fraction of queries in this case, and English is more dominant. Some European languages still occur, but others hardly do (e.g., Polish). Chinese is slightly more prominent compared to other languages, with more than 100,000 queries, but this is still less than 0.1% of all robotic queries.

An interesting observation from Figure 3 is that some languages seem to be "trending" in that we see a clear increase in their popularity. This is particularly strong for Polish, but can also be seen for Catalan. The next section discusses methods that can help to understand such observations.

6.3 Causal Analysis

We have argued that mere statistical analysis can easily be misleading. Indeed, it is not useful to compute averages over exponential distributions (skewed data), which tend to appear on many scales in usage analysis. A better understanding can be obtained by a more fine-grained analysis that tries to attribute the observed traffic to individual causes. Beyond superficial statistics, this gives us valuable insights into the goals underlying current usage.

We have conducted such an analysis by distinguishing queries that are issued by recognisable tools. This analysis started from a number of self-identifying tools, which include the tool's name in a comment in their SPARQL queries.[7] Many further tools were identified from their distinctive traffic patterns, usually marked by bursts of many similar queries, in combination with their user agents. This was mostly manual work, based on inspecting the extracted query patterns that were most frequent, as well as their temporal distribution.

Some of the identified sources have vanished in the following month. For the intervals analysed in this paper, we know of 95 query sources that together account for 144,787,485 (68.39%) of all valid queries. As mentioned in Section 4, we consider three self-identified tools as organic; the respective numbers of queries are: 57,564 (WikiShootMe), 46,691 (SQID), and 996 (Histropedia).

In addition, we identified a browser-based tool that looked up information on local sites based on geographic coordinates in Poland, with a total of 48,051 queries. This activity was notable only in I3 and could be traced back to I2, but not to I1. The sudden prominence

[7] The convention for doing this in the Wikidata community is to start a comment with `#TOOL:` followed by some identifying string.

of the tool is connected to the *Wiki Loves Monuments* competition of Wikipedia, which ran in September 2017, and indeed the respective queries account for the occurrence of property P2186 in Table 8. Moreover, the sudden rise of this new application largely explains why I3 shows somewhat different characteristics in Table 6, and why Polish is seeing a strong upwards trend in Figure 3.

The remaining 91 sources were considered robotic. The top three of them together issue 60.06% of all queries that the Wikidata SPARQL query service has answered in our data. They are:

- *auxiliary matcher*: a data integration tool that identifies potential matches with external datasets; 63,745,842 queries
- *PBB_core fastrun*: a data integration tool for protein databases; 41,605,446 queries
- *bot2*: a multi-purpose bot operated by Magnus Manske;[8] 22,951,459 queries

Further very active query sources include several unidentified scripts and bots. Prominent tools with more than 1M queries query movie database ids, information on association football, and detailed name and label information. The most active sources also include a query that polls the time of the most recent update of the RDF database – this single query has been answered 1,861,752 times (i.e. about once every 11.5 seconds).

Our analysis strongly supports our conjecture that any non-discriminative analysis of the data is necessarily skewed towards a small number of tools. Indeed, both *auxiliary matcher* and *bot2* are maintained by Magnus Manske, which means that he controls about 41% of all traffic. We believe that this is no unusual situation that only occurs for Wikidata, although few individuals will have the impact of a Magnus Manske.

7 ANONYMISATION AND PUBLICATION

We intend to publish our log datasets together with this study, pending the outcome of an ongoing clearance process within the Wikimedia Foundation. Since SPARQL logs are user access logs, they must be treated very carefully to avoid privacy concerns. The published logs therefore are to contain neither IP addresses nor user agents, but they will be partitioned according to our methodology to allow studies of different user groups. They will also contain time stamps for each query.

The queries will also need to be modified to remove any potentially sensitive information. All comments will be removed and each query will be parsed, normalised, and re-serialised. Longer strings, variable names, and geographic coordinates will be replaced uniformly by generic fillers of the same type. We do not expect strings or variable names to contain private information, but the query volume is too large to be certain. Geographic coordinates need to be replaced since they may contain very specific location information that could be linked to individual humans. In contrast, we think that dates (which in Wikidata are only used up to the precision of a day) do not contain enough information to be sensitive. In the current data release proposal, all references to Wikidata entities, and the exact structure of each SPARQL query would be preserved, so that the published files should be of use to other researchers.

Until the anonymised data sets have been released, researchers who wish to replicate our findings can gain access to the data by applying for a formal research collaboration with the Wikimedia Foundation, subject to signing suitable non-disclosure agreements. The source code of our analysis scripts is publicly available.[9] In particular, this code also includes our manual classification rules for queries based on their shape and user agent.

8 CONCLUSION

We have presented a first detailed study of the access logs of the Wikidata SPARQL query service. Due to the varied uses of SPARQL services, the overall query traffic can roughly be classified into two parts: a large fraction of relatively simple queries that are generated by only a few bots, and a much smaller fraction of more complex queries posed (directly or indirectly) by many human users. We argued that, if we want to gain meaningful insights from query log analysis, the queries of the many should outweigh the queries of the few, since the former can more reliably represent a real human information need.

We proposed a characterisation of both components of *organic* and *robotic* traffic, and we showed that these concepts are workable in that (1) both parts can be separated with little effort using a small number of simple signals, and (2) the resulting datasets do indeed mirror many of the expected characteristics. We then continued to study three specific aspects of Wikidata's usage: the practical use of complex RDF encodings, the global imbalance in the use of queries, and the explanation of individual traffic components by means of identifying their sources.

Our research motivates further studies of real SPARQL queries that apply our methods to other datasets. It would be extremely interesting to learn how much organic query traffic can be found in other datasets, and whether it is as distinct from the rest of the queries as it was in our case. Moreover, a causal analysis of the main robotic query sources could shed more light on the actual real-world use of RDF datasets published via SPARQL. Are data integration and systematic download also the predominant tasks in other scenarios? Which fraction of bots can account for which fraction of the traffic? How do more complicated metrics, such as treewidth [3] or specific shapes of query patterns [9], behave for organic and robotic data?

Finally, it would also be of interest to extend the tools available for classifying organic traffic in the first place. Our approach based on user agents, query shapes, and time was feasible but still mostly manual. An automated classification could be of interest, and our data may serve for training and evaluation. Moreover, the approach we took suggests that even anonymised query logs should offer some general user agent information (such as "browser or not"), and also sufficiently fine-grained temporal information. If enough information is retained, then the manual interaction might be avoided completely, allowing the creation of applications that continuously monitor organic traffic for relevant trends. Methodologically speaking, we do indeed seem to be at the very beginning of this field.

Acknowledgements. This work was partly supported by the DFG within the cfaed Cluster of Excellence, CRC 912 (HAEC), and Emmy Noether grant KR 4381/1-1.

[8] https://en.wikipedia.org/wiki/Magnus_Manske

[9] https://github.com/Wikidata/QueryAnalysis

REFERENCES

[1] François Belleau, Marc-Alexandre Nolin, Nicole Tourigny, Philippe Rigault, and Jean Morissette. 2008. Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. *J. of Biomedical Informatics* 41, 5 (2008), 706–716.

[2] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia – A crystallization point for the Web of Data. *J. of Web Semantics* 7, 3 (2009), 154–165.

[3] Angela Bonifati, Wim Martens, and Thomas Timm. 2017. An Analytical Study of Large SPARQL Query Logs. *Proceedings of the VLDB Endowment* 11 (2017), 149–161. Issue 2.

[4] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. 2003. Reasoning on regular path queries. *SIGMOD Record* 32, 4 (2003), 83–92.

[5] Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. 2014. Introducing Wikidata to the Linked Data Web. In *Proc. 13th Int. Semantic Web Conf. (ISWC'14) (LNCS)*, Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandečić, Paul T. Groth, Natasha F. Noy, Krzysztof Janowicz, and Carole A. Goble (Eds.), Vol. 8796. Springer, 50–65.

[6] Daniela Florescu, Alon Levy, and Dan Suciu. 1998. Query containment for conjunctive queries with regular expressions. In *Proc. 17th Symposium on Principles of Database Systems (PODS'98)*, Alberto O. Mendelzon and Jan Paredaens (Eds.). ACM, 139–148.

[7] Steve Harris and Andy Seaborne (Eds.). 21 March 2013. *SPARQL 1.1 Query Language*. W3C Recommendation. Available at http://www.w3.org/TR/sparql11-query/.

[8] Lucie-Aimée Kaffee, Alessandro Piscopo, Pavlos Vougiouklis, Elena Simperl, Leslie Carr, and Lydia Pintscher. 2017. A Glimpse into Babel: An Analysis of Multilinguality in Wikidata. In *Proc. 13th Int. Symposium on Open Collaboration (OpenSym'17)*, Lorraine Morgan (Ed.). ACM, 14:1–14:5.

[9] Mark Kaminski and Egor V. Kostylev. 2018. Complexity and Expressive Power of Weakly Well-Designed SPARQL. *Theory of Computing Systems* (2018), 1–38. https://doi.org/10.1007/s00224-017-9802-9 to appear.

[10] Johannes Lorey and Felix Naumann. 2013. Detecting SPARQL Query Templates for Data Prefetching. In *Proc. 10th Extended Semantic WebConf. (ESWC'13) (LNCS)*, Philipp Cimiano, Óscar Corcho, Valentina Presutti, Laura Hollink, and Sebastian Rudolph (Eds.), Vol. 7882. Springer, 124–139.

[11] Markus Luczak-Roesch, Zamil Aljaloud Saud, Bettina Berendt, and Laura Hollink. 2016. USEWOD 2016 Research Dataset. (2016). https://eprints.soton.ac.uk/385344/

[12] Knud Möller, Michael Hausenblas, Richard Cyganiak, Siegfried Handschuh, and Gunnar A. Grimnes. 2010. Learning from Linked Open Data Usage: Patterns & Metrics. In *Proc. Web Science Conf. (WebSci'10)*.

[13] François Picalausa and Stijn Vansummeren. 2011. What are real SPARQL queries like?. In *Proc. Int. Workshop on Semantic Web Information Management (SWIM'11)*, Roberto De Virgilio, Fausto Giunchiglia, and Letizia Tanca (Eds.). ACM, 6.

[14] Aravindan Raghuveer. 2012. Characterizing Machine Agent Behavior through SPARQL Query Mining. In *Proc. 2nd Int. Workshop on Usage Analysis and the Web of Data (USEWOD'12)*. usewod.org.

[15] Laurens Rietveld and Rinke Hoekstra. 2014. Man vs. Machine: Differences in SPARQL Queries. In *Proc. 4th USEWOD Workshop on Usage Analysis and the Web of Data*. usewod.org.

[16] Muhammad Saleem, Muhammad Intizar Ali, Aidan Hogan, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. 2015. LSQ: The Linked SPARQL Queries Dataset. In *Proc. 14th Int. Semantic Web Conf. (ISWC'15), Part II (LNCS)*, Marcelo Arenas, Óscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul T. Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab (Eds.), Vol. 9367. Springer, 261–269.

[17] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM* 57, 10 (2014).