LOD Lab: Experiments at LOD Scale

Laurens Rietveld[(✉)], Wouter Beek, and Stefan Schlobach

Department of Computer Science,
VU University Amsterdam, Amsterdam, The Netherlands
{laurens.rietveld,w.g.j.beek,stefan.schlobach}@vu.nl

Abstract. Contemporary Semantic Web research is in the business of optimizing algorithms for only a handful of datasets such as DBpedia, BSBM, DBLP and only a few more. This means that current practice does not generally take the true *variety* of Linked Data into account. With hundreds of thousands of datasets out in the world today the results of Semantic Web evaluations are less generalizable than they should and — this paper argues — can be. This paper describes LOD Lab: a fundamentally different evaluation paradigm that makes algorithmic evaluation against hundreds of thousands of datasets the new norm. LOD Lab is implemented in terms of the existing LOD Laundromat architecture combined with the new open-source programming interface *Frank* that supports Web-scale evaluations to be run from the command-line. We illustrate the viability of the LOD Lab approach by rerunning experiments from three recent Semantic Web research publications and expect it will contribute to improving the quality and reproducibility of experimental work in the Semantic Web community. We show that simply rerunning existing experiments within this new evaluation paradigm brings up interesting research questions as to how algorithmic performance relates to (structural) properties of the data.

1 Introduction

While the exact size of the Web of Data is unknown, there is broad agreement that the volume of data published according to Linked Open Data (LOD) standards has to be counted in tens, if not hundreds, of billions of triples by now, originating from hundreds of thousands of datasets from various domains and provenance. This amount and broadness of information makes the Web of Data ideal for testing various types of algorithms and an exciting object of study. As this is widely recognized it is no surprise that many research papers have been published in the recent past that use parts of this enormous and rich collection.

Unfortunately, true large-scale evaluation, both in terms of volume and variety have proven to be much harder to come by than one would expect. One of the main reasons for this is the heterogeneity and user-unfriendliness of the most wide-spread dissemination strategy for Linked Data today: datadumps. Most researchers and application programmers will recognize the problem of dealing with various serialization formats and juggling with syntax errors as well as other data document-specific idiosyncrasies. With the core research being

M. Arenas et al. (Eds.): ISWC 2015, Part II, LNCS 9367, pp. 339–355, 2015.
DOI: 10.1007/978-3-319-25010-6_23

on algorithms and evaluations, data collection, cleaning and harmonization can easily become a barrier too high to overcome.

To avoid these tedious and painful efforts of integrating hundreds of thousands of heterogeneous datasets most current studies with evaluations focus on data published through APIs, e.g., using SPARQL. Although this often provides high-volume datasets for testing, this leads to a strange imbalance in current practice: of the hundreds of thousands of available datasets [15], only around 260 are available through live query endpoints [6], and of the latter less than 10% dominate the evaluation landscape (see Section 2). As such, question-marks have to be put on the generalizability and maybe even validity of many of the results.

Two technological developments of the recent year have changed the situation significantly, though. First, the LOD Laundromat [3], a platform that cleans, harmonizes and republishes Linked Data documents, now serves more than 37 billion triples from over 650,000 data documents in a single, uniform and standards-compliant format. By (re)publishing very many datasets in exactly the same, standards-compliant way, the LOD Laundromat infrastructure supports the evaluation of Semantic Web algorithms on large-scale, heterogeneous and real-world data. In [15] the LOD Laundromat, which had been serving static clean data files until that point, was combined with the Linked Data Fragments (LDF) paradigm [19], thereby offering live query access to its entire collection of cleaned datasets through Web Services (http://lodlaundromat.org).

While these Web Services provide a good interface for some use cases, e.g. downloading a specific data document, the large-scale evaluation of a Semantic Web algorithm against thousands of data documents is still relatively time consuming. This is why we present LOD Lab: an integrated approach towards running Linked Data evaluations in the large. The LOD Lab approach is implemented by pairing the LOD Laundromat backend with *Frank*, an open-source[1] and simple yet flexible front-end programming interface for conducting large-scale experiments over heterogeneous data.

Since the LOD Lab approach defaults to running Semantic Web evaluations against hundreds of thousands of data documents, it introduces a problem that would have been considered a luxury problem even two years ago: now that 650,000 datasets are available, choosing suitable ones for specific experiments becomes a non-trivial task. Fortunately, *Frank* facilitates informed selection by filtering on domain vocabularies and by using metadata about the scraping and cleaning process as well as metadata about the structural properties of the data.

This paper makes the following contributions:

- A new way of conducting Linked Data experiments that incorporates both volume and variety while at the same time allowing the set of considered data documents to be limited according to structural constraints. The motivation for this novel approach is given in Section 2.

[1] See https://github.com/LODLaundry/Frank

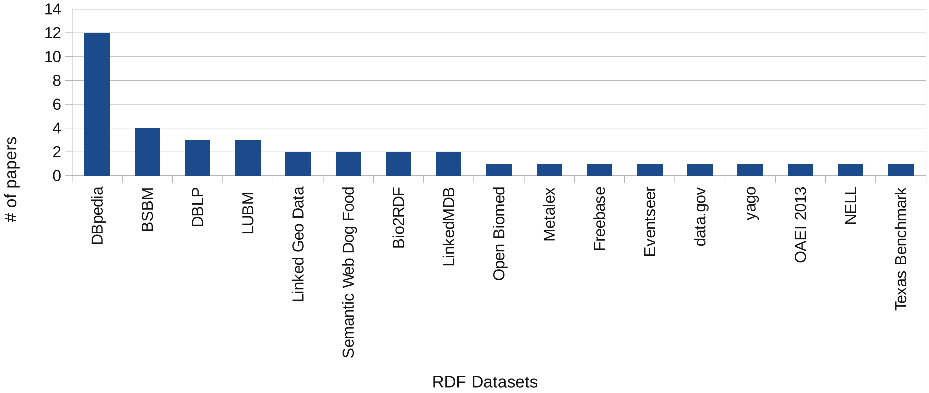


Fig. 1. Overview of datasets used in evaluations of papers accepted in the ISWC 2014 research track. For each dataset the number of articles that use it is shown.

- The introduction of a simple yet versatile programming interface called *Frank* for running large-scale Linked Data evaluations from the command-line. Section 4 discusses the usage, functionality and implementation of *Frank*.
- A demonstration of the viability of the LOD Lab evaluation approach by rerunning three experiments reported in recent Semantic Web conference publications, but now by using hundreds of thousands of data documents. The experiments are described in Section 5.

2 Motivation

Figure 1 gives an overview of the datasets that are used in 20 papers that were accepted in the ISWC 2014 research track. It only includes papers that evaluate Linked Datasets, excluding ones that evaluate algorithms on relatively small ontologies, non-RDF datasets or streamed data. The figure shows that 17 datasets are used in total. The number of datasets per article varies between 1 and 6 and is 2 on average.

The figure shows that most evaluations are conducted on only a handful of datasets. Even the total collection of datasets that are used in these 20 papers is not very large. This implies that many papers evaluate against the same datasets, most often DBpedia. This means that it is generally unclear to what extent published results will transfer to other datasets, specifically those that are only very rarely evaluated against. This is the problem of the *generalizability* of Semantic Web research results (Problem 1).

Problem 1. *By using very few datasets in scientific evaluations, the generalizability of Semantic Web research results is often unknown.*

The reason for Problem 1 is that current evaluation practice does not scale over heterogeneous data, i.e. we face a problem of *variety*. The problem is no longer with the *volume* of the data since most of the datasets that are never evaluated against are smaller than some of the datasets that are currently used in evaluations. While it is sufficiently easy to obtain, load and evaluate one dataset, contemporary practice shows that it is still difficult to do the same thing for very many datasets.

One critique that may be leveled against our identification of Problem 1 is that the most often used datasets are evaluated most often and that evaluation practice is simply in line with data usefulness or relevance. However, most of the algorithms and approaches that are evaluated in Semantic Web research target generic applicability. Specifically, none of the above 20 papers claims to develop a *dataset-specific* approach. Moreover, that a dataset is popular does not imply that results obtained over it are indicative of Linked Data in general and can be transfered to other datasets. This is specifically true for Linked Data where the expressiveness of the language allows datasets to differ considerably.

Empirical surveys have documented the restricted state of today's Semantic Web deployment.[6,12] Many datasets are only available as data dumps, lack dereferenceable URIs, cannot be downloaded due to HTTP errors, cannot be unpacked due to archive errors, or cannot be loaded into Semantic Web tools due to syntax errors. These idiosyncrasies imply in practice that the human costs to run experiments usually increases linearly with the number of datasets. This implies that eager researchers can use one, two, or even six datasets in their evaluations. There is no way, though, to expect hundreds, thousands or even hundreds of thousands of datasets in their evaluations. This lack of variety is due to the fact that the use of every single dataset requires some manual operations (and often repeatedly very similar operations) in order to overcome the aforementioned idiosyncrasies (Hypothesis 1).

Hypothesis 1. *The main reason why experiments are run on very few datasets is that for every dataset a certain amount of manual labor is needed.*

If Hypothesis 1 is correct, then the solution to Problem 1 is to make the human cost of using datasets independent from the number of datasets that is used (Solution 1). The human cost involved in evaluating against datasets should not only be independent of the number of datasets, but should also be low. Both these features can be achieved by fully automating the tasks of obtaining, loading, and using datasets. The LOD Laundromat [3] solves this problem by providing a fully automated infrastructure for disseminating heterogeneous datasets in a unifom and standardized format. It (re)publishes data as cleaned datadumps and, more recently, through Web Services. Neither method is suitable for large-scale evaluation, which requires tools support for fetching, selecting and application of custom algorithms over the appropriate subset of datasets from the LOD Laundromat.

Solution 1. *Make the human effort needed to obtain, load, and use a collection of datasets independent from the size of the collection.*

While running more evaluations against hundreds of thousands of datasets will increase the generalizability of Semantic Web approaches, it also creates a new problem: selectivity (Problem 2). Not every evaluation needs to be, should be nor can be performed on all the available datasets published through the LOD Laundromat. So the question arises which datasets to choose.

Problem 2. *There are currently no means to select those datasets that are pertinent to a given algorithm or approach based on properties of the data.*

The ability to select datasets based on properties of the data also relates to another problem. It is well known, and supported by our results in Section 5 that evaluation outcomes sometimes differ radically for different datasets. Even though this is an interesting observation in itself, it is more pertinent to inquire as to *why* and *how* performance differs over datasets. This is a topic that has traditionally not been touched upon very often in the context of Semantic Web evaluations. LOD Lab will radically simplify future studies in the Semantic Web community to gain insight in how the performance of Semantic Web approaches relates to properties of the data (Problem 3).

Problem 3. *Current evaluations do not relate evaluation outcomes such as the performance of the evaluated algorithm or approach to properties of the data.*

The solution to Problems 2 and 3 is to allow datasets to be selected based on various criteria (Solution 2). These criteria should include a dataset's metadata (e.g., when it was crawled) and structural properties of the data (e.g., the number of unique triples it contains).

Solution 2. *Allow datasets to be selected based on their properties, including the dataset metadata, and structural properties of the data.*

3 Related Work

3.1 Evaluation Frameworks and Benchmarks

Evaluation frameworks and benchmarks have played an important role in Semantic Web research. Many of the previous efforts focused on evaluation of storage and query answering, e.g., in the area of RDF processing and SPARQL query answering, such as the Berlin Benchmark [4], SP^{2}Bench [17], LUBM [9] and Fedbench [18] or LDBC[5]. Those benchmarks usually provide datasets and corresponding query sets, in order to level the playing field and allow for a fair comparisons between tools. Such approaches are a useful source for particular Linked Data research areas. However, most of these approaches present a static or even synthetic dataset. LOD Lab differs from the above by allowing experiments over an extremely high percentage of the real datasets that were published.

Relevant is the Ontology Alignment Evaluation Initiative [7] (OAEI) which presents datasets, and gold standards to relate results to, and a framework for doing so. Most importantly, the OAEI has been using the SEALs[2] evaluation

[2] http://www.seals-project.eu/

platform for years now. SEALs supports experiments on ontology alignment with similar functionality as the LOD Lab supports scalability analytic experiments over multiple various heterogeneous data sources.

3.2 Dataset Collections

The most common large dataset collection to date is a Linked Data crawl published as the Billion Triple Challenge [11] (BTC). The key goal of the Billion Triple Challenge is *'to demonstrate the scalability of applications, as well as the capability to deal with the specifics of data that has been crawled from the public web'.* BTC has indeed proven to facilitate such research, and it has been used in a wide range of papers. The latest BTC dataset collection was published in 2012, and contains 1.4 billion triples. But lets be frank: where this volume used to be 'large', it has now suffered from inflation and is superseded by several larger datasets. Additionally, BTC suffers from the same idiosyncrasies found in other parts of the LOD Cloud: several BTC files contain a sizable number of duplicates and serialization errors[3]. Although the BTC has proven successful for testing algorithms for 'large' data, it lacks the meta-data for dealing with variety: neither dataset characteristics or detailed crawling provenance are available.

Another collection of datasets is LODCache, a Linked Data crawl published via a SPARQL endpoint, exposing (at the time of writing) 34.5 billion triples. Though an interesting source of data, the limitations that the endpoint imposes makes extracting and downloading these datasets difficult. Additionally, no information is published on the crawl mechanism behind it, and the web service lacks meta-data of both the crawl and datasets as well. I.e., this service provides data in a large volume, but lacks the meta-data to *select* datasets.

3.3 Collecting Data on Scale

Some resort to crawling Linked Data themselves considering the lack of available dataset collections. A common tool for this approach is LDspider [13], a Linked Data crawler which supports a wide range of RDF serialization formats, and traverses the Linked Data cloud automatically. This approach requires a large seed list of dataset locations, considering an automatic crawl would need many dereferenceable URIs to automatically discover new datasets. Therefore, LDspider is suitable for some, but crawling larger parts of the LOD Cloud both requires manual effort for curating the seed list, as well as a significant hardware investment.

4 Implementation

LOD Laundromat provides a wealth of data, including the corresponding meta-data such as crawling provenance and structural properties of data documents.

[3] See http://lodlaundromat.org/resource/c926d22eb497788382ffc87a5942f7fb3

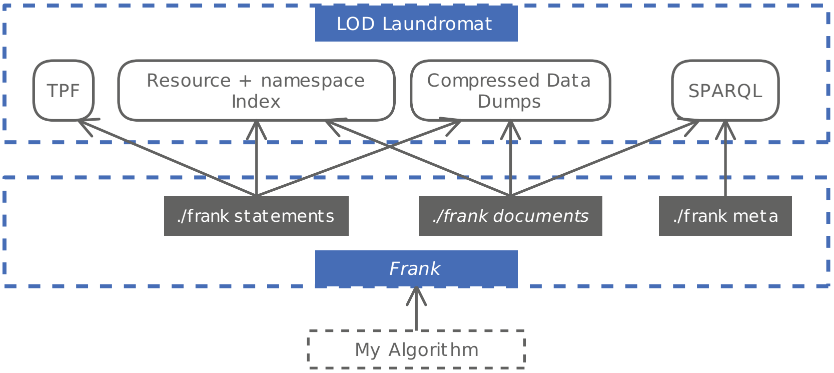


Fig. 2. The implementation architecture for *Frank* and its dependencies on the LOD Laundromat Web Services.

The latter are disseminated through a public SPARQL endpoint[4]. LOD Laundromat data can be accessed by writing a custom script that queries the metadata endpoint to fetch pointers to the relevant data documents. Those pointers either give access to the complete data document or to the Linked Data Fragment API for that particular document. The problem with this approach is that a user needs to be acquainted with the scraping and structural metadata schema used by LOD Laundromat. Since the latter is quite elaborate, designed with versatility rather than usability in mind, the Web Services do not implement Solution 1.

We therefore introduce *Frank*[5], a Bash interface that makes it easy to run evaluations against very large numbers of datasets. By implementing *Frank* in Bash it can be used by all except Windows users who do not want to install Cygwin[6]. Since *Frank* is a plain text file it requires no installation and no inclusion in a software repository or app store, nor does it depend on a specific programming paradigm. As with any Bash script, in- and output can be straightforwardly piped from/to other programs and scripts.

Frank implements Solution 1 since it allows evaluations over hundreds of thousands of data documents to be run by typing a single command (see Section 5 for the commands that were use to scale-up existing experiments). *Frank* implements Solution 2 by offering mechanisms to select datasets according to their metadata, and structural properties (see below for the concrete properties that are supported).

Below, we discuss the three main features of *Frank*: streamed triple retrieval, streamed document retrieval, and metadata retrieval.

[4] See http://lodlaundromat.org/sparql

[5] A technical overview of *Frank* was presented at the ESWC Developers Workshop [2].

[6] See https://www.cygwin.com/

4.1 Streamed Triple Retrieval

`frank statements` allows individual atomic statements or triples to be retrieved. When called without arguments this streams all 37 billion triples by fetching and unpacking the Gzipped LOD Laundromat data dumps. If called with the command-line flags `--subject`, `--predicate`, and/or `--object`, only triples that contain the specified subject-, predicate- and object-term are returned. These three flags mimic the expressivity of the Linked Data Fragment (LDF) [19] Web API. They are expressively equivalent to SPARQL queries with a single-line Basic Graph Pattern (BGP) [10]. LDF supports streamed processing though a self-descriptive API that uses pagination in order to serve large results in smaller chunks. If called with a subject, predicate and/or object flag, `frank statements` interfaces with the LOD Laundromat index[7] which contains a mapping between all LOD Laundromat resources and documents. For these documents, *Frank* connects with the Linked Data Fragments API for, handling the LDF pagination settings in order to ensure a constant stream of triples. The LDF API is able to answer triple pattern requests efficiently by using the Header Dictionary Triples[8] (HDT) technology. HDT is a binary, compressed and indexed serialization format that facilitates efficient browsing and querying of RDF data at the level of single-line BGPs. HDT files are automatically generated for all data documents that are disseminated by the LOD Laundromat backend.

4.2 Streamed Document Retrieval

`frank documents` allows individual documents to be retrieved. The command interfaces with the SPARQL endpoint and LOD Laundromat index in order to find data documents that satisfy the given properties.

The following selection mechanisms are supported by `frank documents`:

- Flags `--minTriples` and `--maxTriples` filter data documents based on the number of unique triples they contain.
- Filtering on the average minimum and maximum degree (as well as *in* and *out* degree), e.g. `--minAvgDegree`
- Flag `--namespace` connects to the LOD Laundromat namespace index, and only returns documents using that particular namespace. This allows for coarse selectivity of domains. For instance datasets that are possibly relevant to the bioinformatics domain can be filtered based on the `drugbank` and `chebi` namespaces. The namespace flag accepts both full URIs and de-facto RDF prefixes[9] that denote namespaces.
- Flag `--sparql` allows an arbitrarily complex SPARQL query to be evaluated against the LOD Laundromat backend. While not very user-friendly, this flag allows less often used selection criteria to be applied. Since we log SPARQL queries at the backend, we are able to add flags to *Frank* based on often requested queries.

[7] See http://index.lodlaundromat.org

[8] See http://www.rdfhdt.org/

[9] Prefixes are taken from http://prefix.cc.

Data document are identified in the following two ways:

1. The URI from which the data document, cleaned by the LOD Laundromat, can be downloaded (`--downloadUri`). These clean data documents are disseminated by the LOD Laundromat as Gzipped N-Triples or N-Quads. The statements are unique within a document so no bookkeeping with respect to duplicate occurrences needs to be applied. Statements are returned according to their lexicographic order. These statements can be processed on a one-by-one basis which allows for streamed processing by *Frank*.
2. The Semantic Web resource identifier assigned by LOD Laundromat for this particular document (`--resourceUri`).

When neither `--downloadUri` nor `--resourceUri` are passed as arguments *Frank* returns both separated by a white-space.

The streaming nature of *Frank* enables combinations of streamed triple and document retrieval. The following command returns a stream of documents with an average out-degree of 15 that contain at least 100 unique RDF properties. The stream consists of N-Quads where every triple ends in a newline and within-triple newlines are escape according to the N-Quads standard. The graph name of each quadruple is the LOD Laundromat document identifier.

```
$ ./frank documents \
    --resourceUri \
    --minAvgOutDegree 15 \
    --sparql "?doc llm:metrics/llm:distinctProperties ?numProp.
             (FILTER ?numProp > 100)"
  | ./frank statements --showGraph
```

4.3 Metadata

`frank meta` retrieves the metadata description of a given data document. It interfaces with the SPARQL endpoint of LOD Laundromat and returns N-Triples that contain provenance and structural properties for that particular document[10].

These structural properties include:

- VoID description properties such as the number of triples, entities, and the number of used properties and classes
- Additional properties not included in VoID directly, such as the number of *defined* properties and classes, and the number of literals, IRIs, and blank nodes.
- Network properties such as degree, in degree and out degree. For each of these properties we present descriptive statistics including the minimum, maximum, median, mean and standard deviation.
- Details on the IRI and literal lengths, with similar descriptive statistics.

[10] We present this metadata collection in more detail in [14].

Other than such structural properties, the LOD Laundromat metadata includes crawling provenance as well, such as:

- A reference to the original download location of the document
- Warnings and errors encountered when fetching and cleaning the document
- Number of duplicate triples
- Temporal information such as the last-modified date of the original file, or the cleaning date of a document.
- Other low-level information on the original file, such as the serialization format, its size, or its line count

5 Evaluation

To illustrate the use of the LOD Lab for evaluation purposes, we re-evaluate parts of three previously published papers. A paper presenting an efficient in-memory RDF dictionary (Section 5.1), a paper compressing RDF in a binary representations (Section 5.2), and a paper exploring Linked Data best practices (Section 5.3). We do not aim to completely reproduce these papers, as we merely intend to illustrate LOD Lab and how *Frank* can be used by others.

Below we discuss these papers in detail and highlight the parts of their experiment we reproduce. For these experiments we illustrate how we used *Frank*, and we present the reevaluated results. The source-code of these evaluations are publicly available[11]

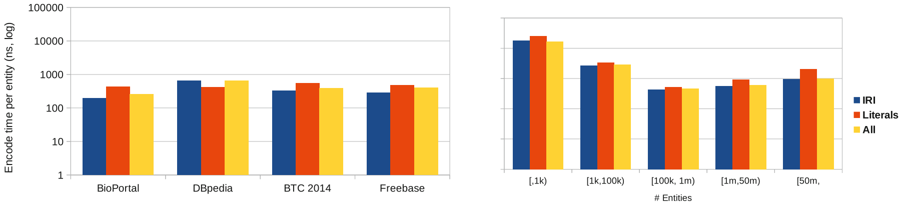
5.1 Paper 1: RDF Vault

'A Compact In-Memory Dictionary for RDF data' [1] is a recent paper from the 2015 Extended Semantic Web Conference, which presents RDF Vault. RDF Vault is an in-memory dictionary, which takes advantage of string similarities of IRIs, as many IRIs share the same prefix. The authors take inspiration from conventional Tries (tree structures for storing data), and optimize this method for RDF data.

The authors measure the average encoding time per entity (time it takes to *store* a string in RDF Vault), average decoding time per entity (time it takes to *get* this string), and the memory use. Additionally, the authors make a distinction between these measurements for literals and URIs, considering literals often lack a common prefix. In the original paper, RDF vault is compared against several baselines (e.g. a classical in-memory dictionary), and evaluated against the following 4 datasets: Freebase, the Billion Triple Challenge datasets, DBpedia and BioPortal.

We use *Frank* to re-evaluate the encoding time of RDF Vault (using the original implementation) against a larger number of datasets: for each document, we measure the average encoding time of literals, IRIs, and both combined. In

[11] See https://github.com/LaurensRietveld/FrankEvaluations



(a) Results from [1]

(b) LOD Lab results

Fig. 3. Average encoding time per entity (ns)

order to compare these results meaningfully with the results from the original paper, we group the documents by number of entities, and present the encoding/decoding time for each group.

In figure 3 we present the original RDF vault results on the left side, and the results obtained via *Frank* on the right side. We collected the results from frank by piping all documents to the evaluation script as follows, where `./rdfVaultEncodeDocument.sh` is a Bash script that reads the *Frank* documents from the standard input, and applies RDF Vault for each of these documents.

```
$ ./frank documents --downloadUri | ./rdfVaultEncodeDocument.sh
```

Both figures show the average encoding time of IRIs, Literals, and both combined. Our results are based on 100,000 LOD Laundromat documents[12], where we grouped documents in buckets by the number of encoded entities. The original results differ between datasets: the average encoding time of IRIs in BioPortal are 1/3 of the DBpedia encoding times. Our results show the influence of the dataset size on the encoding times (particularly considering the y log scale). Smaller datasets of less than 1,000 entities may take up to 30.000 nano seconds per entity. Similarly, datasets with between 1,000 and 100,000 entities show longer encoding times than the original paper as well. For dataset sizes which correspond to the original paper, the results are similar. The re-evaluation of these results clearly show the effect of the dataset size on encoding times. That effect was not investigated in the original paper, because the experiments were only done on a handful of datasets. As we have shown, *Frank* trivially allows to run the original experiments on hundreds of thousands datasets, immediately giving an insight in the unexpected non-monotonic relation between dataset size and encoding time per entity.

Other structural dimensions might be relevant for this paper as well, such as the number of literals in a dataset or the standard deviation of URI or literal lengths. All these dimension are accessible using the LOD Laundromat metadata and the *Frank* interface. E.g., to run the vault experiments for dataset with a high standard deviation in URI lengths, run:

[12] Due to the runtime of RDF Vault and time constraints we were unable to re-evaluate this on the complete LOD Laundromat set.

```
$ ./frank documents \
  --downloadUri \
  --query "{?doc llm:metrics/llm:IRILength/llm:std ?std .
             FILTER(?std >  50)}"
    | ./rdfVaultEncodeDocument.sh
```

5.2 Paper 2: RDF HDT

'Binary RDF Representation for Publication and Exchange (HDT)' [8] is an often cited paper (56 at the time of writing) from the journal of Web Semantics. HDT is a compact binary RDF representation which partitions RDF datasets in three components: Header information, a dictionary, and the actual triples structure. The important gain of HDT is that the HDT files are queryable *in their compressed form* using simple SPARQL triple patterns.

In the original paper, the performance of HDT is evaluated by measuring the compression ratio of HDT compared to other compression algorithms (e.g. Gzip and Bzip2), the compression time, and by measuring the number of entries in the dictionary compared to the total number of triples. The datasets used in this evaluation are Geonames, Wikipedia, DBTune, Uniprot and DBpedia-en. A part of the evaluation is evaluated against the 400 largest datasets in the Billion Triple Challenge (BTC). This is a fairly complete evaluation, considering the number of datasets, and the use of BTC datasets.

The results we re-evaluate[13] are the compression ratios presented in [8] which were evaluated on Uniprot datasets from different sizes (1, 5, 10, 20, 30 and 40 million triples). We re-evaluate this particular research result using *Frank* by finding dataset of similar sizes (± 10%) and by measuring the compression ratio.

The LOD Laundromat documents are fetched using *Frank* and filtered to match the Uniprot dataset sizes. E.g., to select LOD Laundromat documents matching the 1 million Uniprot dataset, *Frank* searches for documents of 1 million with a deviation of 10%, and streams these document to a shell script which downloads and compresses these documents using HDT.

```
$ ./frank documents --minTriples 950000 --maxTriples 1050000
     | ./hdtCompressDocument.sh
```

Table 1 shows the compression ratio for Uniprot datasets on the left side, and the average compression ratio for LOD Laundromat documents on the right side. There is a large difference between Uniprot and the LOD Laundromat datasets in both compression ratio and average document size. Another interesting observation is the high average compression ratio of LOD Laundromat documents around 1 million, compared to other LOD Laundromat documents.

To better understand such differences, we use *Frank* to evaluate RDF HDT along another dimension: the average degree of documents. We did so by searching for three buckets of datasets. Those with a low (1-5), medium (5-10) and high (10+) average degree, all with at least 1 million triples:

[13] We re-evaluated the latest HDT version accessible at https://github.com/rdfhdt/

Table 1. HDT Compression rates: Results from [8] on Uniprot (left side) vs. results from *Frank* (right side)

	Original: Uniprot			**LOD Lab**		
Triples (millions)	# docs	Size (MB)	Compression Ratio	# docs	Avg. Size (MB)	Avg Compression Ratio
1	1	89.07	3.73%	179	183.31	11.23%
5	1	444.71	3.48%	74	799.98	4.99%
10	1	893.39	3.27%	50	1,642.60	5.43%
20	1	1,790.41	3.31%	17	3,328.57	4.15%
30	1	2,680.51	3.27%	19	4,880.26	5.09%
40	1	3,574.59	3.26%	8	6,586.95	7.25%

Table 2. HDT Compression rates grouped by avg degree

Avg. Degree	# docs	Compression Ratio
1-5	92	21.68%
5-10	80	6.67%
10-∞	99	4.85%

```
$ ./frank documents --minAvgDegree 5 --maxAvgDegree 10 --minTriples 1000000
     | ./hdtCompressDocument.sh
```

The results (See Table 2) show that an increase in degree of a document comes with a decrease in compression ratio.

These experimentation on a large numbers of datasets across a large number of dimensions is made easy by *Frank*, and allows researchers to both tune their algorithms to different document characteristics, as well as better understand their algorithms behavior under different conditions.

5.3 Paper 3: Linked Data Best Practices

Other than using the LOD Lab for empirical evaluations, we show how it can be used for explorative and observational papers as well. The most cited paper of the International Semantic Web Conference 2014 is 'Adoption of the Linked Data Best Practices in Different Topical Domains' [16], where the authors analyze Linked Data best practices by crawling (using LDspider [13]) the LOD Cloud. Seed items for this crawl come from public catalogs, the Billion Triple Challenge, and datasets advertised on public LOD mailing lists. The crawl included 900,129 documents (URIs that were dereferenced) and 8,038,396 resources. Documents are grouped to 1014 datasets using information from catalogs, or Pay-Level-Domain (PLD) otherwise. The paper present a large and diverse set of statistics, including:

1. The number of resource per document
2. Dataset grouped by topical domain. These domains are fetched from online catalogs if any, and manually annotated otherwise
3. Indegree and outdegree of datasets
4. The links occurring between datasets, and the type of predicates used for linking
5. The use of vocabularies in datasets

The crawling mechanism behind these statistics strongly relies on dereferenceable URIs. As a consequence, there is a strong link between a crawled document and the URI it is crawled from: we know which URI is the 'authority' for a document. This offers opportunities for e.g. grouping the datasets by PLD and finding links between datasets. This crawling mechanism differs from the LOD Laundromat, which mostly consists of (often compressed) data dumps[14]. As a consequence, in LOD Laundromat, the URL http://data.dws.informatik.uni-mannheim.de/dbpedia/2014/en/ (the official DBpedia download location) does not directly match with http://dbpedia.org/resource/Amsterdam, making it difficult to know the authoritativeness of the download dump URI. I.e., the LOD Laundromat crawls many more documents and triples (including those not accessible as dereferenceable URI), but lacks information on the authoritativeness of URIs. Vice versa, the used crawl in [16] crawls only a fraction of the LOD Laundromat size, but retains the notion of authority. As a result, the original paper has statistics on DBpedia as a whole, where the LOD Lab results are separate for each independent DBpedia data dump.

These differences in features between both crawling mechanisms restricts the ability of *Frank* to reproduce all of the statistics from [16]. However, we chose to focus on re-evaluating the used vocabularies on the LOD Cloud, which does not suffer from these difference in crawling mechanisms. Instead, *Frank* offers a more complete perspective on the use of vocabularies, considering the number of crawled triples.

We reproduced this experiment by simply streaming all the LOD Laundromat download URIs to a script counting the namespaces[15]:

```
$ ./frank documents --downloadUri | ./countNamespacesForDocument.sh
```

Table 3 shows the 10 most frequent occurring namespaces in documents. In the original paper these counts are grouped by dataset (i.e. groups of documents), where we present these statistics on a document level alone.

This table shows striking differences: where the *time* namespace is used in 68.20% of the LOD Laundromat documents, it does not occur in the top 10 list of [16]. Similarly, the *cube* namespace occurs in 23.92% of LOD Laundromat documents, and is missing from the original top 10 list as well.

The crawling method behind both approaches, and the method used by [16] to group documents as datasets can explain these discrepancies. Therefore, we

[14] See [3] for more information.

[15] Using the namespace list of http://prefix.cc, similar to the original paper.

Table 3. Top 10 namespaces used in documents

Original [16]			LOD Lab		
Prefix	#datasets	% datasets	Prefix	#docs	% docs
rdf	996	98.22%	rdf	639,575	98.40%
rdfs	736	72.58%	time	443,222	68.19%
foaf	701	69.13%	cube	155,460	23.92%
dcterm	568	56.01%	sdmxdim	154,940	23.84%
owl	370	36.49%	worldbank	147,362	22.67%
wgs84	254	25.05%	interval	69,270	10.66%
sioc	179	17.65%	rdfs	30,422	4.68%
admin	157	15.48%	dcterms	26,368	4.06%
skos	143	14.11%	foaf	20,468	3.15%
void	137	13.51%	dc	14,423	2.22%

do not claim to have the right answer for these kind of statistics. Instead, we show that the LOD Lab approach allows for large scale comparisons for these kinds of Linked Data observational studies.

6 Conclusion

The distributed nature of the Semantic Web, the wide range of serialization formats, and the idiosyncrasies found in datasets, make it difficult to use the Semantic Web as a true large-scale evaluation platform. As a consequence, most research papers are only evaluated against a handful of datasets.

In this paper we presented LOD Lab, a new way of conducting Linked Data experiments that incorporates both volume and variety while at the same time allowing the set of considered data documents to be limited according to domain-specific and/or structural constraints. This is achieved by using the LOD Laundromat backend together with the simple yet versatile programming interface *Frank* that allows large-scale Linked Data evaluations to be run from the command-line.

The viability of the LOD Lab approach was demonstrated by scaling up three experiments reported in recent Semantic Web conference publications. These re-evaluations show that evaluations over Linked Data can now be performed without the human effort having to increase linearly in terms of the number of datasets involved. In addition, the re-evaluations show that the combination of volume, variety and selectivity facilitates a more detailed analysis of Semantic Web algorithms and approaches by relating evaluation outcomes to properties of the data.

References

1. Bazoobandi, H.R., de Rooij, S., Urbani, J., ten Teije, A., van Harmelen, F., Bal, H.: A compact in-memory dictionary for RDF data. In: Gandon, F., Sabou, M., Sack, H., d'Amato, C., Cudré-Mauroux, P., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9088, pp. 205–220. Springer, Heidelberg (2015)
2. Beek, W., Rietveld, L.: Frank: The lod cloud at your fingertips. In: Developers Workshop, ESWC (2015)
3. Beek, W., Rietveld, L., Bazoobandi, H.R., Wielemaker, J., Schlobach, S.: LOD laundromat: a uniform way of publishing other people's dirty data. In: Mika, P., Tudorache, T., Bernstein, A., Welty, C., Knoblock, C., Vrandečić, D., Groth, P., Noy, N., Janowicz, K., Goble, C. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 213–228. Springer, Heidelberg (2014)
4. Bizer, C., Schultz, A.: The berlin sparql benchmark (2009)
5. Boncz, P., Fundulaki, I., Gubichev, A., Larriba-Pey, J., Neumann, T.: The linked data benchmark council project. Datenbank-Spektrum **13**(2), 121–129 (2013)
6. Buil-Aranda, C., Hogan, A., Umbrich, J., Vandenbussche, P.-Y.: SPARQL web-querying infrastructure: ready for action? In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) ISWC 2013, Part II. LNCS, vol. 8219, pp. 277–293. Springer, Heidelberg (2013)
7. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology alignment evaluation initiative: six years of experience. In: Spaccapietra, S. (ed.) Journal on Data Semantics XV. LNCS, vol. 6720, pp. 158–192. Springer, Heidelberg (2011)
8. Fernández, J.D., Martínez-Prieto, M.A., Gutiérrez, C., Polleres, A., Arias, M.: Binary rdf representation for publication and exchange (hdt). Web Semantics: Science, Services and Agents on the World Wide Web **19**, 22–41 (2013)
9. Guo, Y., Pan, Z., Heflin, J.: Lubm: A benchmark for owl knowledge base systems. Web Semantics: Science, Services and Agents on the World Wide Web **3**(2), 158–182 (2005)
10. Harris, S., Seaborne, A.: SPARQL 1.1 query language, March 2013
11. Harth, A.: Billion Triples Challenge data set. Downloaded from (2012). http://km.aifb.kit.edu/projects/btc-2012/
12. Hogan, A., Umbrich, J., Harth, A., Cyganiak, R., Polleres, A., Decker, S.: An Empirical Survey of Linked Data Conformance. Web Semantics: Science, Services and Agents on the World Wide Web **14**, 14–44 (2012)
13. Isele, R., Umbrich, J., Bizer, C., Harth, A.: Ldspider: an open-source crawling framework for the web of linked data. In: 9th International Semantic Web Conference (ISWC 2010). Citeseer (2010)
14. Rietveld, L., Beek, W., Schlobach, S.: LOD in a box: The C-LOD meta-dataset (Under submission). http://www.semantic-web-journal.net/system/files/swj868.pdf
15. Rietveld, L., Verborgh, R., Beek, W., Vander Sande, M., Schlobach, S.: Linked data-as-a-service: the semantic web redeployed. In: Gandon, F., Sabou, M., Sack, H., d'Amato, C., Cudré-Mauroux, P., Zimmermann, A. (eds.) ESWC 2015. LNCS, vol. 9088, pp. 471–487. Springer, Heidelberg (2015)

16. Schmachtenberg, M., Bizer, C., Paulheim, H.: Adoption of the linked data best practices in different topical domains. In: Mika, P., Tudorache, T., Bernstein, A., Welty, C., Knoblock, C., Vrandečić, D., Groth, P., Noy, N., Janowicz, K., Goble, C. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 245–260. Springer, Heidelberg (2014)
17. Schmidt, M., Hornung, T., Lausen, G., Pinkel, C.: Sp 2 bench: A sparql performance benchmark, icde, Shanghai, China (2009)
18. Schmidt, M., Görlitz, O., Haase, P., Ladwig, G., Schwarte, A., Tran, T.: FedBench: a benchmark suite for federated semantic data query processing. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 585–600. Springer, Heidelberg (2011)
19. Verborgh, R., et al.: Querying datasets on the web with high availability. In: Mika, P., et al. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 180–196. Springer, Heidelberg (2014)