Estimating query rewriting quality over the LOD

Editor(s): Name Surname, University, Country
Solicited review(s): Name Surname, University, Country
Open review(s): Name Surname, University, Country

Ana I. Torre-Bastida [a,*], Jesús Bermúdez [b] and Arantza Illarramendi [b]
[a] *Tecnalia Research & Innovation*
Spain
E-mail: isabel.torre@tecnalia.com
[b] *Basque Country University UPV-EHU*
Spain
E-mail: jesus.bermudez@ehu.es, a.illarramendi@ehu.es

Abstract. Nowadays users have difficulty to query datasets with different vocabularies and data structures that are included in the Linked Data environment. For this reason it is interesting to develop systems that can produce on demand rewritings of queries. Moreover, a semantics preserving rewriting cannot often be guaranteed by those systems due to heterogeneity of the vocabularies. It is at this point where the quality estimation of the produced rewriting becomes crucial. Notice that, in a real scenario, there is not a reference query.

In this paper we present a novel framework that, given a query written in the vocabulary the user is more familiar with, the system rewrites the query in terms of the vocabulary of a target dataset. Moreover, it also informs about the quality of the rewritten query with two scores: firstly, a similarity factor which is based on the rewriting process itself, and so can be considered of intensional nature; and secondly, a quality estimation that can be considered of extensional nature offered by a predictive model. This model is constructed by a machine learning algorithm that learns from a set of queries and their intended (gold standard) rewritings.

The feasibility of the framework has been validated in a real scenario.

Keywords: Semantic web, RDF, Linked Open Data, SPARQL, Query rewriting, Similarity

1. Introduction

The increasing adoption of the Linked Open Data (LOD) paradigm has generated a distributed space of globally interlinked data, usually known as the Web of Data. This new space opens up the possibility of executing exploratory and selective queries over a huge set of updated data. However, many users find difficulties when formulating queries over it, due to the fact that they are not familiar with the data, links and vocabularies of many heterogeneous datasets that constitute the Web of Data. In this scenario it becomes necessary to provide the users with tools and mechanisms that help them to exploit the vast amount of available data.

We can find in the specialized literature different proposals that have considered the goal of providing distributed query processing mechanisms, where queries are evaluated against the distributed datasets of the Web of Data. Among them, two main groups can be distinguished: those that follow the *federated query* processing approach (e.g. [10]) in which a query against a federation of datasets is split into queries that can be answered in the individual nodes where datasets

[*]Corresponding author. E-mail: isabel.torre@tecnalia.com

are stored. And those that follow the *exploratory query* processing approach (e.g. [11]), where a query is first evaluated on an initial dataset and then the Web of Data is explored by traversing interesting links pointing to other datasets stored in different nodes which may contain more data entities satisfying the query. The work presented in this paper follows this second approach.

Although the majority of exploratory query processing approaches consider semantic-preserving navigation among datasets such as [18], our proposal deals with a scenario where the preservation of the semantics is not a strong requirement and therefore it considers semantics-preserving and non-semantics-preserving navigations in order to increase the opportunities of getting results. When a non-semantics-preserving scenario is considered, the definition of a quality estimation of the rewritten query becomes crucial because the user needs to be aware of the confidence that can be deposited on the results obtained from the new dataset.

As a motivating example, let us imagine a user that is only familiar with the LinkedMDB SPARQL endpoint (a dataset about movies and their related people). This user asks for the names of art directors working on the films directed by Woody Allen. The SPARQL query constructed by the user could be the following one:

```
SELECT  DISTINCT ?name
WHERE {
  ?woody  movie:director_name "Woody Allen".
  ?movie  movie:director ?woody;
  movie:film_art_director ?art.
  ?art movie:film_art_director_name ?name. }
```

and the obtained results are listed on table 1:

?name
“Tom Warren"

Table 1

Query results from LinkedMDB.

Moreover the user would find useful to execute the same query in other datasets, perhaps more recognized ones or more active ones, trying to obtain more results. One good example of those datasets can be DBpedia. Using our proposal the user could obtain the following reformulation of the query, according to the DBpedia vocabulary:

```
SELECT  DISTINCT ?name
  WHERE {
  ?woody  foaf:name "Woody Allen"@en.
  ?movie  dbo:director ?woody;
  dbo:cinematography ?art.
  ?art foaf:name ?name. }
```

This reformulation has been based on some declared mappings between both datasets: `movie:director` was declared an equivalent term to `dbo:director`, and properties `movie:director_name` and `movie:film_art_director_name` were declared as subproperties of `foaf:name`. Although no declared mapping for `movie:film_art_director` existed, the system proposed the term `dbo:cinematography` as an approximation to the original one. The obtained results are listed on table 2:

?name
“Sven Nykvist"
“Zhao Fei"
“Harris Savides"
“David M. Walsh"
“Remi Adefarasin"
“Carlo Di Palma"
“Javier Aguirresarobe"
“Gordon Willis"
“Vilmos Zsigmond ASC"
“Darius Khondji"
“Ghislain Cloquet"

Table 2

Query results from DBpedia.

Notice that new results appeared when querying the DBpedia dataset which may be of interest to the user. Nevertheless, which further enriches the answer is the provision of a quality score of the reformulated query. It is at this point where a main contribution of this paper plays a relevant role.

In general, quality estimation can be defined in terms of a query similarity measure between the source and target queries (source query, formulated over the initial dataset, and target query, formulated over another dataset of the Web of Data indicated as target). However, when comparing two queries, different similarity dimensions can be considered [5]: (1) the query structure, expressed as a string or a graph structure; (2) the query content, its triple patterns and ontological terms and literal values; (3) the language features, such as query operators and modifiers; and (4) the results set retrieved by the query. Queries may be assessed with respect to one or several of that considered dimensions. And it is widely accepted that the application context heavily determines the choice for a similarity measure. In the scenario considered in this paper we think that the content and the result set are the appropriate dimensions because structure, or language of the target query are irrelevant to the user that formulates the query over the source dataset. Notwithstanding that the result set is what matters to the user. It is crucial to notice that the intention of issuing the query to a target dataset is to look for more or different results than those obtained by the source query. Therefore, the intended result set of the target query cannot be compared with that of the source query in terms of exact matching and the query similarity measure must take into account this distinctive feature.

The main contribution of this paper is twofold: (1) Proposal of a general framework for the deployment and management of a query rewriting system concerning the scenario previously explained. And (2) the validation of the framework in a real context, including the machine learning and optimization techniques used to estimate the quality of the rewriting outcome.

Proposal of a general Framework We propose a new framework which considers two kind of users:

– *End users*, who select or formulate a query over a source dataset. And the system issues a new query, which mimics the original one, over a target dataset (of the Web of Data) with the goal of enriching the results obtained from the source dataset. The new issued query is annotated with a quality estimation composed by a prediction of the widely known information retrieval metric F1 score, and a similarity factor between queries.
– *Expert/technical users*, who in addition to benefiting from the functionalities provided for end users, can also include in the framework: rewriting rules, an algorithm to process them, and similarity measures to qualify the query rewriting. The framework also provides them with facilities to tune the introduced similarity measures by means of optimization techniques. The rewriting rules, similarity measures and training queries introduced are stored in the log of the framework with the idea of serving as experimental and comparison benchmark.

Validation of the Framework The framework has been tested in a real context. For that, we have filled the framework with the following elements:

– Query rewriting rules. Apart from some rules dealing with the rewriting of terms by their specified equivalents, via synonym mappings or EDOAL (Expressive and Declarative Ontology Alignment Language) [8] alignment rules, the framework also deals with some other heuristic based rules which conform a carefully controlled set of cases.
– Similarity measures. The computation of the quality estimation takes into account different similarity measures depending on the motif of the rule being applied. Those motifs range from relational to ontological structure, and from language based to context based similarity.
– Datasets. Three domain areas were considered for the datasets: media-domain, bibliographic, and life science. From the media-domain six datasets were selected, five datasets from the bibliographic domain, and five more from the life science domain, respectively.
– Queries. 100 queries were formulated over the previously selected datasets. When selecting the queries, our aim was to get a set that would contain a broad spectrum of SPARQL query types [1]. Concerning provenance we selected queries that appeared in well known benchmarks such as QALD4 or FedBench, and we also considered queries that belonged to LOD SPARQL endpoints logs from the selected datasets.

The rest of the paper is organized as follows. Section 2 presents some related works in the scope of resource matching and query rewriting. Section 3 introduces an abstract framework for our proposal. Section 4 shows a framework embodiment. Section 5 describes the framework validation results. Finally, some conclusions are presented.

2. Related works

The impressive growth of the Web of Data has pushed the research on Data Linking [21]: "the task of determining whether two object descriptions can be linked one to the other to represent the fact that they refer to the same real-world object in a given domain or the fact that some kind of relation holds between them". Those object descriptions can be expressed with diverse structural relationships, depending on different contexts, using classes and properties from different ontologies. Research on object similarity and class matching has issued a considerable amount of techniques and systems on the field of Ontology Matching [7], although less work has been devised to property alignment [17,2]. A recent work [16] presents an unsupervised learning process for instance matching between entities. Queries considered in this paper involve terms for classes, properties and individuals. Therefore, techniques for discovering similarity for any of them are relevant. However, the topic of this paper regards query similarity, which can be recognized as a different problem. As has been noticed in [5], the appropriate notion of query similarity depends on the goal of the task. In our case, the task is to estimate the similarity of the intended semantics between a query designed for a source dataset and a rewriting to a different vocabulary, to be evaluated in a different target dataset.

Some works, for example [18,3], have approached a restricted version of the task carried out in our case. They restrict themselves to produce semantics preserving translations (i.e. total similarity) and so they assume that enough equivalent correspondences exist among entities in datasets. Taking into account that such an assumption is too strong in real scenarios we consider situations where different types of correspondences exist (not only of equivalence type) and even more, situations where some correspondences are missing. This consideration implies that query semantics is sometimes not preserved in the rewriting process and therefore the estimation of similarity of the produced rewriting becomes crucial.

The aim of our considered rewriting is to look for more answers in a target dataset than those obtained from the source dataset. Some other works have the goal of obtaining more answers (including approximate ones) for an original query; however, all of them restrict their scope to a single source dataset. All those works can be situated under the topic of query relaxation. In [15] they propose a logical relaxation of conditions in conjunctive queries based on RDFS semantics. Those conditions are successively turned more general and a ranking in the successively obtained answers is generated. [13,14] use the same kind of relaxations as [15], but propose different ranking models. In [13], similarity of relaxed queries is measured with a model based on the distance between nodes in the ontology hierarchy. In [14], they use an information content based model to measure similarity of relaxed queries. The work in [6] addresses the query relaxation problem by broadening or reformulating triple patterns of the queries. Their framework admits replacement of terms by other terms or by variables and also removal of entire triple patterns. In that work, generation and ranking of relaxed queries is guided by statistical techniques: a distance between the language models associated to entity documents is defined.

With different use cases in mind, the papers [12, 4,20] present different possibilities for approaching the querying of Linked Data. In [12] a framework for relaxation of star-shaped SPARQL queries is proposed. They present different *matchers* (functions that map pairs of values to a relaxation score) for different kinds of attributes (numeric, lexical or categorical). The framework may involve multiple matchers. The matchers generate a tuple of numeric distances between a query and an entity (answer for the query). Notice that the distance is defined between an entity and a query, not between two queries as in our approach. [4] proposes a measure to evaluate the similarity between a graph representing a query and a graph representing the dataset. With a suitable relaxation of the notion of alignment between query graph paths and dataset graph paths they generate approximate answers to queries. In [20] a method for query approximation, query relaxation, and their combination is proposed for providing flexible querying capabilities that assist users in formulating queries. Query answers are ranked in order of increasing distance from the user's original query.

In summary, cited works that transform the query or reformulate the notion of answer in order to provide users with more answers from the source dataset, do not try to reformulate the query in a different dataset with a priori unknown vocabulary; and this is a distinguishing feature of our use case.

3. Abstract framework

An abstract representation of the proposed framework for rewriting a query and estimating the quality of the rewritten query, can be expressed as an structure ($\mathcal{R}$, $\mathcal{A}$, $\mathcal{M}$, $\mathcal{V}$, $\mathcal{SF}$, $\mathcal{P}$) where

- $\mathcal{R}$ is a set of SPARQL query rewriting rules,
- $\mathcal{A}$ is the algorithm for applying the rules,
- $\mathcal{M}$ is a set of similarity measures between fragments of query expressions,
- $\mathcal{V} : \mathcal{R} \rightarrow \mathcal{M}$ is an application that associates a similarity measure to every rule,
- $\mathcal{SF} : \mathcal{R}^* \rightarrow [0, 1]$ which associates each sequence of applied rewriting rules with a similarity factor from the $[0, 1]$ real interval, and
- $\mathcal{P}$ is a predictive model which estimates a quality score for the target query.

The rule language to express rules in $\mathcal{R}$ is similar to CONSTRUCT expressions of the SPARQL 1.1 language. The query patterns in the WHERE clause describe the conditions that must be matched to apply a rule and the CONSTRUCT clause describe the pattern which will replace the rewritten fragment of the current query. Algorithm $\mathcal{A}$ manages the rewriting process, the order in which rules are applied and so on.

Every application of a rule r is considered as a step in the progress to the target query, and such step is valuated with a factor computed by the associated similarity measure $\mathcal{V}(r)$ from $\mathcal{M}$. The rewriting system of the proposed framework takes a given query Q_s (named *source query*), expressed with a vocabulary adequate for the source dataset, and transforms it into another query Q_t (named *target query*), expressed with a vocabulary adequate for the selected target dataset. The rewriting process produces Q_t as a semantically equivalent query to Q_s as long as enough equivalence mappings between the vocabulary of the source dataset and the vocabulary of the target dataset are found. But the distinguishing point is that the process produces a mimetic query Q_t even in the case when no equivalent translation for Q_s is found. That is to say, semantic preservation cannot be guaranteed due to vocabularies heterogeneity and/or missing links with terms appearing in the source query. It is at this point where the definition of a quality estimation of the rewriting outcome becomes crucial to our approach, because the user needs to be aware of the quality of the produced target query.

The function $\mathcal{SF}$ calculates a similarity factor for a target query in terms of the sequence of rules $\bar{r}$ that were applied to construct it and combining properly the measures $\mathcal{V}(r)$ (for each $r \in \bar{r}$). Similarity measures in $\mathcal{M}$ can be defined by simple functions or very complex ones. Usually they can be defined by combining similarity measures taken from a state-of-the-art repository [7]. Moreover, optimization methods algorithms can be used to tune parameter values for the linear combination of measures, taking advantage of the experimental scenario.

As previously said in the introduction section, the intention of issuing a query to the target dataset is to look for more or different results than those obtained in the source dataset. Therefore, although the target query should try to maintain the spirit of the source query, the intended result set of the target query cannot be compared with the source query retrieved ones but with that of an *ideal* expression of such source query in terms of the vocabulary acceptable by the target dataset. Notice that, due to the previously mentioned heterogeneity reasons, such ideal expression cannot be trivially constructed. In fact, we consider that the finding of such ideal expression, in the considered scenario, should be realized by a human expert who knows vocabularies of source and target datasets. And, therefore, the reference query against which the target query should be compared is a human designed one, that tries to express the most similar intention to the source query but in the context of the target dataset. We consider such a query our *gold standard* query against which the target query should be compared.

In the presence of a gold standard query, its results can be compared with those obtained by the target query. Statistical measures such as precision, recall and F1 score can be used to measure the quality of a target query. Of course, gold standards can only exist in an experimental scenario but not in the real setting, and that's the reason to incorporate machine learning techniques in the framework. The predictive model $\mathcal{P}$ is generated by a supervised machine learning method applied to a suitable experimental scenario: selected benchmark of source queries with their respective gold standard queries for the target datasets, corresponding target queries generated by the rewriting system with their respective $\mathcal{SF}$ value and with their respective score that will be the goal for prediction.

4. Framework embodiment

This section presents a brief explanation of a specific embodiment of the abstract framework ($\mathcal{R}$, $\mathcal{A}$, $\mathcal{M}$, $\mathcal{V}$, $\mathcal{SF}$, $\mathcal{P}$) that has been partially presented in [23] and which serves as a proof of concept for our proposal.

A SPARQL query can be represented by a graph pattern consisting on a set of *triple patterns*. A *triple pattern* is a triple (s, p, o) where s is the subject, p is the predicate, and o is the object. The three of them represent resources and any of them can be a *variable* (denoted by prefixing it with a question mark, for instance $?x$).

The set of rules $\mathcal{R}$ was devised from a pragmatic point of view. The rules set up a common sense heuristics to obtain acceptable rewritings even when no semantically equivalent translations are at hand. Preconditions for the application of the rules take into account a carefully restricted context of the terms occurring in the graph pattern. Although restricted, the rule set has shown to be quite effective achieving acceptable rewritings (see section 5).

Five kinds of rules have been considered (see table 3), each kind based on a different motif: Equivalence (E), Hierarchy (H), Answer-based (A), Profile-based (P), and Feature-based (F). A pragmatic scenario has been considered in which a bridge dataset can be taken into account in the process of rewriting a query adequate for a source dataset into another query adequate for a target dataset. In order to favour the possibilities of finding alignments between resources, mappings between both the source (D_s) and target (D_t) datasets and a bridge (D_b) dataset are considered. That scenario is quite frequent, since in almost any domain there is a popular dataset that may play such a reference role.

In table 3, the first column (*Rule*) shows the kind of the rule and a rule number. The second column (*To be replaced*) shows a generic triple pattern that should be matched to a triple pattern in the current query, such pattern will be replaced by the graph pattern in the fourth column (*Replacement*) if patterns in the third column (*Knowledge Base*) are present in the knowledge base graph of the datasets in question. Prefixes *s:*, *t:*, and *b:* refers to source, target, and bridge datasets respectively. Finally, the fifth column ($\mathcal{V}(r)$) presents a similarity measure definition for the corresponding rule. The measure is meant to reflect a similarity value between the replaced part and the replacement after application of the rule. Rules are applied to a triple pattern of the query being rewritten that presents a resource identifier (i.e. URI) u that does not belong to the target dataset vocabulary. Therefore, each application of a rule can be associated to the *unknown* resource u that was replaced; and that is why the similarity measure in the $\mathcal{V}(r)$ column is represented by a function in terms of u, whose value $\phi(u)$ is a real number in the interval $[0, 1]$.

The algorithm $\mathcal{A}$ for applying the rules is quite straightforward in this case. It consists in applying the rules in the same sequence order that they are numbered in table 3. The idea behind this order is to apply first those rules that seems to maintain as much as possible the semantics of the original query. A rule is applied as long as its preconditions (described by columns *To be replaced* and *Knowledge Base*) are satisfied. When a rule is no longer applicable, the algorithm drives to the following rule. Something specific takes place when application of rule F15 has finished. If any non adequate URI remains in the query, rules E and H are tried again and after that, any triple pattern presenting a non adequate URI is deleted from the query.

The similarity measure associated to equivalence rules (E) is simply the constant function $\phi(u) = 1$. In contrast, the similarity function associated to hierarchy rules (H) is quite more complex. It is an adaptation of a distance proposed in [25] and elsewhere for a distance between two terms in a hierarchy. Each term U in the hierarchy is associated with a *milestone* value *m(U)* depending on its depth in the hierarchy. Then, the distance between two terms U and V in the hierarchy is $d(U, ccp(U,V)) + d(V, ccp(U,V))$ where *ccp(U,V)* is the *closest common parent* of U and V in the hierarchy and $d(X, ccp(X,Y)) = m(ccp(X,Y)) - m(X)$. However, in the context considered in this paper, where U and V belong to different vocabularies and, therefore, different hierarchies, the notion of *ccp(U,V)* is not directly applicable. Then, we have adapted such distance in the following manner. The intuition is that the deepest term (i.e. that with the least *milestone*) carries more information than the higher term. The value of parameter k is the decrease factor for the milestone in every next level of the hierarchy. It is $k = 2$ in our case.

$$m(U) = m(V) \wedge U \sqsubseteq V \rightarrow$$
$$distance(U,V) = m(U) \times (1 - \frac{1}{k})$$
$$m(U) < m(V) \wedge U \sqsubseteq V \rightarrow$$
$$distance(U,V) = m(V) - m(U)$$
$$m(U) < m(V) \wedge V \sqsubseteq U \rightarrow$$
$$distance(U,V) = m(U) \times (1 - \frac{1}{k})$$

Once the distance is defined, the similarity function S_o between U and V is

$$S_o(U,V) = 1 - distance(U,V)$$

Depending on the particular instantiation of the applied rewriting rule, the number of terms involved in the replacement of the term u and its triple pattern can be more than one (see table 3); in fact, $\text{AND}_{i=1,\dots,k}$ (t: u_{1i}, u_2, u_3) or $\text{UNION}_{i=1,\dots,k}$ (t: u_{1i}, u_2, u_3). In such a case, the similarity measure is the average of the k pairwise similarity values:

$$\phi(u) = \frac{\sum_{i=1}^{k} S_o(u, u_i)}{k}$$

With respect to answer-based (A), profile-based (P), and feature-based (F) rules, the similarity measure is defined as a linear combination of some other three

Rule	**To be replaced**	**Knowledge Base**	**Replacement**	$\mathcal{V}(r)$
E1	ELHS	$EDOAL : ELHS \rightarrow t : ERHS$	$t : ERHS$	$\phi(u_i) = 1$
E2	(u_1, u_2, u_3)	$(u_1, eq, t : u_{1i})$ $(i = 1, \ldots, k)$ Analogously for u_2 and u_3	$\text{UNION}_{i=1,\ldots,k}$ $(t : u_{1i}, u_2, u_3)$	
E3	(u_1, u_2, u_3)	$(u_1, eq, b : u_{1i})(b : u_{1i}, eq, t : u_{1i})$ $(i = 1, \ldots, k)$ Analogously for u_2 and u_3	$\text{UNION}_{i=1,\ldots,k}$ $(t : u_{1i}, u_2, u_3)$	
H4	(u_1, u_2, u_3)	(u_1, sub, v) $(v, sub, t : u_{1i})$ $(i = 1, \ldots, k)$ Analogously for u_2 and u_3	$\text{AND}_{i=1,\ldots,k}$ $(t : u_{1i}, u_2, u_3)$	$\phi(u_1) = \frac{\sum_{i=1}^{k} S_o(u_1, t\!: u_i)}{k}$
H5	(u_1, u_2, u_3)	$(t : v_{1i}, sub, v)$ (v, sub, u_1) $(i = 1, \ldots, k)$ Analogously for u_2 and u_3	$\text{UNION}_{i=1,\ldots,k}$ $(t : v_{1i}, u_2, u_3)$	
A6	$(?x, t : p, u)$	Answers$(?x, t : p, u)$= $(x_1, \ldots, x_n)$ $(x_k, eq, t : x_k)$ $(k = 1, \ldots, n)$ $(t : x_k, t : p, t : o_{kj})$ $(j = 1, \ldots, m_k)$	$\text{UNION}_{k=1,\ldots,n}($ $\text{AND}_{j=1,\ldots,m_k}$ $(?x, t : p, t : o_{kj})\ )$	$\phi(u) =$ $\alpha_n \cdot S_n(u, t\!: b)$ $+\alpha_d \cdot S_d(u, t\!: b)$ $+\alpha_o \cdot S_o(u, t\!: b)$
A7	$(u, t : p, ?x)$	Analogous to 6, subject instead of object		
A8	$(?x, p, t : o)$	Anwers$(?x, p, t : o)$= $(x_1, \ldots, x_n\ )$ $(x_i, eq, t : x_i)$ $(i = 1, \ldots, n)$ $\forall j \in \{1 \ldots k\}$ $(t : x_i, t : p_j, t : o)$	$\text{UNION}_{j=1,\ldots,k}$ $(?x, t : p_k, t : o)$	
A9	$(t : s, p, ?x)$	Analogous to 8, subject instead of object		
A10	$(?x, ?p, o)$	Answers$(?x, ?p, o)$= $(x_1, \ldots, x_n)$ $(x_k, eq, t : x_k)$ $(k = 1, \ldots, n)$ $(t : x_k, t : p_{ki}, t : o_{ki})$ $(i = 1, \ldots, m)$ $t : o_z =$ mostFrequent$(t : o_{ki} :$ $k = 1, \ldots, n.i = 1, \ldots, m)$	$(?x, ?p, t : o_z)$	
A11	$(s, ?p, ?x)$	Analogous to 10, subject instead of object		
P12	(u, p, o)	$(u, s : p_i, a_i)$ $(a_i, eq, t : a_i)$ $(t : a_i, t : p_i, t : o_i)$ $(i = 1, \ldots, m)$ $(b_j, s : q_j, u)$ $(b_j, eq, t : b_j)$ $(t : b_j, t : q_j, t : o_j)$ $(j = m + 1, \ldots, n)$ $t : o_z$= maxSim $(u, t : o_1, \ldots, t : o_n)$	$(t : o_z, p, o)$	$\phi(u) =$ $\alpha_n \cdot S_n(u, t\!: b)$ $+\alpha_d \cdot S_d(u, t\!: b)$ $+\alpha_o \cdot S_o(u, t\!: b)$
P13	(s, p, u)	Analogous to 12, object instead of subject		
P14	(s, p, o)	Analogous to 12, predicate instead of subject		
F15	(u_1, u_2, u_3)	$(u_1, s : p_k, s : o_k)$ $(k = 1, \ldots, n)$ Analogously for u_2 and u_3	$(?v, u_2, u_3)$ $\text{AND}_{k=1,\ldots,n}$ $(u_1, s : p_k, s : o_k)$ $?v$ a new variable	$\phi(u) =$ $\alpha_n \cdot S_n(u, t\!: b)$ $+\alpha_d \cdot S_d(u, t\!: b)$ $+\alpha_o \cdot S_o(u, t\!: b)$

Table 3
SPARQL query rewriting rules and similarity measures.

similarity measures, which are selected to cover various facets of comparison between terms. Assuming that the term u is replaced by the term $t\colon b$, the similarity measure is

$$\phi(u) = \alpha_n \cdot S_n(u, t\colon b) + \alpha_d \cdot S_d(u, t\colon b) + \alpha_o \cdot S_o(u, t\colon b)$$

$$\alpha_n + \alpha_d + \alpha_o \geq 0 \;\wedge\; \alpha_n + \alpha_d + \alpha_o = 1$$

S_n and S_d are string based methods, and S_o is the similarity measure previously defined. S_n is a similarity measure computed as the average of Levenshtein and Jaccard distances, corresponding to the `rdfs:label` property value of the two compared terms. And S_d is a token based similarity measure which takes into account the definition contexts of the terms. For each compared term, u and $t\colon b$, a bag of words is constructed containing words from their `rdfs:comment` and `rdfs:label` string valued properties. S_d is defined as a cosine similarity of two vectors $V(u)$ and $V(t\colon b)$ constructed by the frequency of word appearance (i.e. Vector Space Model technique):

$$S_d(u, t\colon b) = \frac{V(u) \cdot V(t\colon b)}{\|V(u)\| \, \|V(t\colon b)\|}$$

As has been previously noted, when the number of terms involved in the replacement of the term u is more than one (let us say $(t\colon b_1, \ldots, t\colon b_k)$), every single measure is replaced by the corresponding average $(S_x(u, t\colon b_1) + \ldots + S_x(u, t\colon b_k))/k$.

The parameter values α_n, α_d, and α_o can be determined by an expert taking into account the desired pondering of the three facets. But it could be preferable to obtain those parameter values as the output of an specifically designed optimization algorithm. In our case, we leveraged on a genetic algorithm called Harmony Search (HS) for the determination of such parameter values. The explanation of the configuration of the algorithm will be presented in section 5.

The similariry measure $\mathcal{SF}$ associated to a target query is an aggregation of the similarity values associated to each rule applied to reach such a target query. Among different possibilities, a measure based on the Euclidean distance on a n-dimensional space was selected. Given a sequence of rule applications $(r_i)_{i=1}^N$ for the rewriting of a source query into a target query, involving the corresponding non adequate terms $(u_i)_{i=1}^N$, the values $(\phi(u_i))_{i=1}^N$ can be considered the coordinates of a point in a N-dimensional space, where the point $(1, \ldots, 1)$ represents the best and the point $(0, \ldots, 0)$ the worst. Then, the Euclidean distance between the points $(\phi(u_i))_{i=1}^N$ and $(1, \ldots, 1)$ provides a foundation for a similarity measure. In order to normalize the similarity value within the real interval $[0, 1]$, with the value 1 representing the best similarity, the Euclidean distance between $(\phi(u_i))_{i=1}^N$ and $(1, \ldots, 1)$ is divided by $\sqrt{N}$, and substracted from the best similarity 1.

$$\mathcal{SF}((u_i)_{i=1}^N) = 1 - \frac{1}{\sqrt{N}} \sqrt{\sum_{i=1}^{N} (1 - \phi(u_i))^2}$$

Finally, the score selected to inform about the quality of the obtained target query was the F1 score calculated by comparing the answers retrieved by the target query with those retrieved by the corresponding Gold standard query. We call *Relevant answers* (Rel) to the set of answers obtained by running the Gold standard query, and *Retrieved answers* (Ret) to the set of answers obtained by running the target query. Then, the values Precision (P), Recall (R), and F1 score (F1) are calculated with the following formulae.

$$\mathrm{P} = \frac{|\mathrm{Rel} \cap \mathrm{Ret}|}{|\mathrm{Ret}|} \qquad \mathrm{R} = \frac{|\mathrm{Rel} \cap \mathrm{Ret}|}{|\mathrm{Rel}|} \qquad \mathrm{F1} = 2 \times \frac{\mathrm{P} \times \mathrm{R}}{\mathrm{P} + \mathrm{R}}$$

The supervised learning model $\mathcal{P}$ devised to predict the F1 score of a target query was generated from the application of a Random Forest algorithm. Some other regression algorithms were considered and after an experimentation process, discussed in section 5, the Random Forest was selected because it offered the best results.

5. Framework validation

This section presents the main results of the process carried out to validate the proposed framework. The following resources are presented: (a) the LOD datasets selected for querying data, (b) the collection of training queries, (c) the optimization algorithm used to determine the proper parameter values for computing similarity measures and a discussion of its results, (d) the collection of features gathered from data to construct the datasets used by the machine learning algorithms and the results obtained by them.

5.1. Queries and datasets

The set of experimental queries were selected after analyzing heterogeneous benchmarks such as QALD[1], FedBench [22], and real SPARQL endpoint logs (BNE, DBpedia). The idea underlying the selection process was to select queries that could be representative of the different SPARQL query types and that could cover heterogeneous domains in the Linked Open Data framework.

We trusted on well known datasets, with available SPARQL endpoints and accessible for people. Three domain areas were considered for the datasets: media-domain, bibliographic, and life science. For each one, a set of recognized datasets were selected. With respect to media-domain, the selection was: DBpedia, MusicBrainz, LinkedMDB, Jamendo, New York Times and BBC. With respect to bibliographic domain, we considered BNE (Biblioteca Nacional de España), BNF (BibliothÃÍque National du France), BNB (British National Bibliography), LIBRIS, and Cambridge. And finally for the life science area: Drugbank, SIDER, CHEBI, DISEASOME, and KEGG were the selected ones. Moreover, to achieve greater plurality in the tests, we used the SP2Bench, which is based on a synthetic dataset.

A set containing 100 queries was created for experimenting with the framework and providing data for the learning process. Those queries along with their corresponding gold standards and the names of source and target datasets are listed in the appendix hosted in the following URL [2].

Regarding the syntactic structure of the queries, a variety of the SPARQL operators (UNION, OPTIONAL, FILTER) and different schemas for joins of variables appear in the queries. The number of triple patterns of each query ranges from 1 to 7.

5.2. Suitability of the similarity factor

The similarity factor $\mathcal{SF}$ in the framework is intended to inform the user about the expected quality of a target query obtained by the rewriting system. Assuming that the ideal for the target query would be to perform the most similar to the corresponding gold standard query, it is natural to design $\mathcal{SF}$ in such a way that the similarity factor associated to a target query be correlated with the F1 score of that target query. Therefore, tuning of the similarity measures used to compute $\mathcal{SF}$ is desirable.

The similarity measure, presented in section 4, defined as a linear combination of three similarity measures, involves three parameters α_n, α_d, and α_o. Instead of trying to determine their appropriate values by chance it seems preferable to devise a method to optimize their values towards the goal of moving $\mathcal{SF}$ closer to F1 score.

We selected a method based on a genetic algorithm, specifically the Harmony Search (HS) algorithm [9]. Harmonies represent sets of variables to optimize, whereas the quality of the harmony is given by the fitness function of the optimization problem at hand.

In our case the variables to optimize are the parameters $(\alpha_n, \alpha_d, \alpha_o)$ appearing in the definition of our similarity measure, and the established fitness function was the maximization of the proportion of queries whose absolute difference between the value of the similarity factor $\mathcal{SF}((u_i)_{i=1}^N)$ and the F1 score for the corresponding target query is less than a given threshold β. The fitness function is as follows:

$$\begin{aligned} &\text{maximize} && \sum_{i=1}^{n} \frac{1}{n} H(q_i)) \\ &\text{subject to} && 0 \le \alpha_n, \alpha_d, \alpha_o, \beta \le 1 \end{aligned}$$

$$H(q_i) = \begin{cases} 1 & \text{if } |F_1(q_i) - \mathcal{SF}(q_i, \alpha_n, \alpha_d, \alpha_o)| < \beta \\ 0 & \text{otherwise} \end{cases}$$

In order to carry out the optimization process we considered the previously presented query set (see 5.1), and divided it into training and test datasets, with a ratio of 80% and 20%, respectively. The first step was to execute the HS algorithm on the set of training queries, in order to obtain the parameter values that achieve optimal fitness. For this, the algorithm was parameterized with the number of iterations and initial values for the parameters. The HS optimization process may obtain different solutions depending on the initial random values chosen for the parameters and the number of iterations allowed. With 100 iterations and an initialization defined by the HS algorithm itself, we obtained the values shown in table 4 for parameters according to different values of β $(0.4, 0.2, 0.1)$.

Convergence of the HS optimization process for the training dataset and $\beta = 0.2$ is shown in figure 1, where

[1]http://greententacle.techfak.uni-bielefeld.de/ cunger/qald/

[2]`https://github.com/anaistobas/SPARQLQuerySet`

	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.4$
α_n	0.11902	0.13049	0.13405
α_d	0.50183	0.51760	0.53741
α_o	0.37915	0.35191	0.32854

Table 4

Similarity parameter values for training dataset.

abscissas axis represents the number of algorithm iterations and the ordinate axis represents the fitness value. It can be observed how the fitness increases with the number of iterations.

To assess the validity of the parameter values obtained by the algorithm, the similarity factor was computed over the set of 20% test queries (in this case using the alphas obtained in the different scenarios with $\beta = 0.4, 0.2, 0.1$, respectively) and then, the absolute difference between these similarity factors and the F1 scores for the corresponding target queries were calculated. Training and Test Fitness in table 5 display the values for training dataset fitness and test dataset fitness respectively. The relation between the training dataset fitness value and the one obtained with the test dataset measures the suitability of the optimization.

It can be observed that the absolute difference between the fitness obtained with the training and test datasets never exceeds 0.14, which indicates that the optimization process is valid (values less than 0.3 are considered valid).

5.3. Discussion

Following we present the comparison of the similarity factor $\mathcal{SF}$ and F1 score values obtained by our embodied framework using the optimized parameter values calculated by the HS aforementioned method over the set of 100 experimental queries. Figure 2 shows a scatterplot of the 100 points with coordinates (F1 score, $\mathcal{SF}$). Table 6 shows numbers for the same points.

An analysis of the results revealed the following considerations: In 59 out of 100 source queries the target queries provided the same set of results as the corresponding gold standard queries. From this set, in 50 of them their F1 score was 1, and in 9 of them (Q11, Q17, Q25, Q26, Q39, Q41, Q52, Q76, and Q91) the F1 score could not be calculated because the sets of relevant and retrieved results (see section 4) were both empty (notice that eventual dataset updates could change those results). See table 7 for a graphical display of the results.

The cases in which the F1 score equals 1 (50%) can be divided into two groups depending on the similarity factor: (1) Similarity factor equal to F1 score, represent a 23%. (2) Similarity factor is less than F1 score, represents a 27%.

In the other case, there are 41 queries in which the target queries did not provide the same set of results as the corresponding gold standard queries. Therefore these queries have an F1 score lower than 1. From this set, in 24 of them the similarity factor is higher than the F1 score, for example queries Q2 $(0.002, 0.405)$, Q24 $(0.05, 0.44)$, or Q55 $(0.018, 1)$. In those cases the similarity factor is too optimistic because the actual results provided by the target query are very different from those provided by the gold standard query, these are the cases where the F1-measure value is very low. In 15 of them the similarity factor is lower than the F1 score. And finally, there are 2 queries where the retrieved results are empty (Q28, Q40).

We want to notice that in cases where $\mathcal{SF} <$ F1 score, the rewriting system is performing better than the offered similarity factor, since the higher F1 score shows that the target query performs more similarly to the gold standard. It can be argued that while $\mathcal{SF}$ reflects an intensional measure (semantic similarity of the replacement), the F1 score has an extensional character.

We think that the results of table 6 allow us to say that the similarity factor defined in section 4 is quite informative about the quality of the target query from an intensional point of view. Nevertheless, one goal of the presented framework is to serve as a tool for establishing benchmarks which promote improvement of query rewriting systems.

5.4. Predictive model for the F1 score

As we have already mentioned, in a real scenario gold standard queries are not available, that is the reason why a predictive model $\mathcal{P}$ is considered in the framework. In the scenario of this paper, $\mathcal{P}$ is in charge of predicting the F1 score.

The construction of that model is based on datasets that contain features that represent underlying structure and characteristics of the data subject of the prediction. In our scenario, features related to the structure of the source query such as number of triple patterns or number of operators, along with features related to rules that take part during the rewriting process, and finally features concerning the involved LOD datasets, were considered to build the feature datasets.

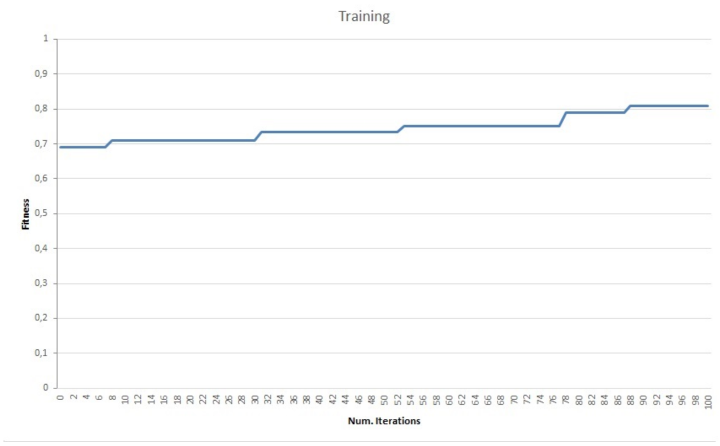

Fig. 1. Convergence of fitness with the training dataset and $\beta = 0.2$.

	Training Fitness	*Test Fitness*
$\boldsymbol{\beta = 0,4}$	0.84	0.77
$\boldsymbol{\beta = 0,2}$	0.81	0.72
$\boldsymbol{\beta = 0,1}$	0.67	0.53

Table 5

Fitness values for different thresholds.

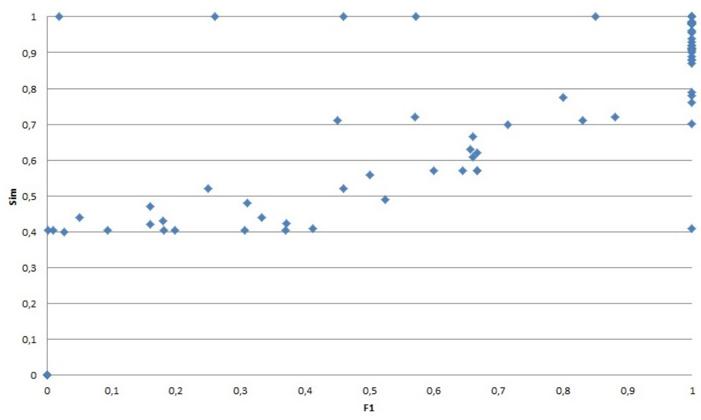

Fig. 2. Scatterplot for F1 score and Similarity factor (using similarity parameter values calculated for training dataset and $\beta = 0.2$).

Following we present the 21 considered features:

1. Similarity and rules features, numbered from 1 to 11: (1) Number of times the equivalence rules are applied, (2) Similarity measure value associated to the equivalence rules application, (3) Number of times the hierarchy rules are applied, (4) Similarity measure value associated to the hierarchy rules application, (5) Number of times the answer-based rules are applied, (6) Similarity measure value associated to the answer-based rules application, (7) Number of times the profile-based rules are applied, (8) Similarity measure value associated to the profile-based rules application, (9) Number of times the feature-based rules are applied, (10) Similarity measure value associated to the feature-based rules application, and (11) the similarity factor calculated for the target query.
2. Query structure features, numbered from 12 to 18: (12) Number of triple patterns of the source query, (13) number of terms of the source query, (14) number of terms not belonging to the vocab-

	Q1	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q9**	**Q10**
F1	1	0.002	1	1	0.83	1	0.46	1	0.5	1
$\mathcal{SF}$	0.956	0.405	0.987	0.979	0.71	0.905	0.52	0.984	0.56	0.78
	Q11	**Q12**	**Q13**	**Q14**	**Q15**	**Q16**	**Q17**	**Q18**	**Q19**	**Q20**
F1	0	1	1	1	1	1	0	1	1	1
$\mathcal{SF}$	0	0.912	1	1	1	1	0	0.901	1	1
	Q21	**Q22**	**Q23**	**Q24**	**Q25**	**Q26**	**Q27**	**Q28**	**Q29**	**Q30**
F1	1	0.66	0.16	0.05	0	0	0.01	0	0.8	1
$\mathcal{SF}$	1	0.61	0.47	0.44	0	0	0.405	0	0.775	0.701
	Q31	**Q32**	**Q33**	**Q34**	**Q35**	**Q36**	**Q37**	**Q38**	**Q39**	**Q40**
F1	1	1	1	0.57	0.88	1	0.18	1	0	0
$\mathcal{SF}$	1	0.79	0.87	0.72	0.72	0.93	0.43	1	0	0
	Q41	**Q42**	**Q43**	**Q44**	**Q45**	**Q46**	**Q47**	**Q48**	**Q49**	**Q50**
F1	0	1	1	0.31	0.37	0.25	1	1	1	0.45
$\mathcal{SF}$	0	0.88	0.41	0.48	0,405	0.52	1	1	0.761	0.711
	Q51	**Q52**	**Q53**	**Q54**	**Q55**	**Q56**	**Q57**	**Q58**	**Q59**	**Q60**
F1	1	0	0.666	1	0.018	0.198	0.666	1	1	1
$\mathcal{SF}$	0.96	0	0.57	0.94	1	0.405	0.665	0.919	0.982	0,89
	Q61	**Q62**	**Q63**	**Q64**	**Q65**	**Q66**	**Q67**	**Q68**	**Q69**	**Q70**
F1	1	0.371	0.306	0.656	0.666	0.714	1	1	1	1
$\mathcal{SF}$	1	0.422	0.405	0.63	0.62	0.7	1	0.88	1	1
	Q71	**Q72**	**Q73**	**Q74**	**Q75**	**Q76**	**Q77**	**Q78**	**Q79**	**Q80**
F1	1	0.666	0.571	0.26	1	0	0.85	1	0.46	1
$\mathcal{SF}$	1	0.57	1	0.99	1	0	1	1	1	1
	Q81	**Q82**	**Q83**	**Q84**	**Q85**	**Q86**	**Q87**	**Q88**	**Q89**	**Q90**
F1	1	1	1	0.026	0.644	0.412	0.181	1	1	0.6
$\mathcal{SF}$	1	0.98	0.91	0.4	0.57	0.41	0.405	1	0.96	0.57
	Q91	**Q92**	**Q93**	**Q94**	**Q95**	**Q96**	**Q97**	**Q98**	**Q99**	**Q100**
F1	0	1	1	0.524	0.093	0.333	1	1	1	0.16
$\mathcal{SF}$	0	0.92	1	0.49	0.405	0.44	1	0.98	0.91	0.42

Table 6

$\mathcal{SF}$ (using similarity parameter values calculated for training dataset and $\beta = 0.2$) and F1 score for the experimental query set.

Query set 100 queries					
Retrieved answers = Relevant answers 59 queries			Ret$\neq$ Rel 41 queries		
F1 = 1 50 queries		F1 cannot be calculated (without answers) 9 queries	$\mathcal{SF}$ > F1 24	$\mathcal{SF}$ < F1 15	|Ret|=0 2
$\mathcal{SF} =$ F1 23 queries	$\mathcal{SF}$ < F1 27 queries				

Table 7

Summary of the comparison between $\mathcal{SF}$ value and F1 score.

		F.1	F.2	F.3	F.4	F.5	F.6	F.7	F.8
1	Equivalence rule application times	X	X						
2	Equivalence rule similarity measure value	X	X	X					
3	Hierarchy rule application times	X	X						
4	Hierarchy rule similarity measure value	X	X	X					
5	Answer rule application times	X	X						
6	Answer rule similarity measure value	X	X	X					
7	Profile rule application times	X	X						
8	Profile rule similarity measure value	X	X	X					
9	Feature rule application times	X	X						
10	Feature rule similarity measure value	X	X	X					
11	Overall similarity measure	X	X	X	X	X	X	X	X
12	Number of source triple patterns	X	X						
13	Number of terms in source query	X	X	X	X	X	X	X	
14	Number of non-adequate terms	X	X	X	X	X	X		X
15	Number of union operators	X							
16	Number of projected variables	X							
17	Number of optional operators	X							
18	Number of filter operators	X							
19	Source Dataset	X	X	X	X				
20	Target Dataset	X	X	X	X				
21	Number of mappings between source and target	X	X			X			

Table 8

Selected features for the different datasets.

Datasets	**LR**	**SVM**	**SVM +PCA**	**RF**	**RF+ PCA**
Features1	0.6751	-1.5072	-1.941	**0.8219**	-1,0013
Features2	0.5902	-1.0313	-3.8922	0.7132	0.2461
Features3	0.6045	-0.0437	-1.5331	0.7152	0.3479
Features4	0.4572	-0.0689	0.0215	0.7026	0.6045
Features5	0.7146	**0.7731**	0.0329	0.6835	0.6218
Features6	**0.7529**	0.6815	**0.1844**	0.7797	0.6414
Features7	0.7511	0.7611	0.1521	0.7221	0.6511
Features8	0.7523	0.7146	0.1711	0.7923	**0.687**

Table 9

R2 metric of the predictive models.

ulary of the target dataset, (15) number of union operators, (16) number of projected variables, (17) number of optional operators, and (18) number of filter operators.

3. LOD Datasets features, numbered from 19 to 21: (19) categorical data associated to the source dataset depending on its size, (20) categorical data associated to the target dataset depending on its size, and (21) number of mappings between source and target datasets.

In order to select a best fit model, we experimented with the following off-the-shelf algorithms [19,24]: Linear regression (LR), Support Vector Machines (SVM) with and without PCA (Principal Component Analysis), and Random Forest with and without PCA; and with 8 different datasets (F.1 to F.8) corresponding to distinct feature selection (see table 8).

The values used in the experiment were obtained from the rewriting of the 100 aforementioned queries. This set of queries were divided in three fragments: 80% for the training process, 15% for the validation process, and 5% for the test process, respectively. The score of each of those models is measured based on a 20-fold cross-validated average mean squared error-R2 metric. The results are presented in the table 9, where each cell of the table represents the coefficient of determination for each model (in combination or not with PCA) trained with the feature dataset indicated by the row.

As can be seen, the model that best fit is that obtained using the Random Forest-RF with F.1 dataset, with a R2 equals 0.8219. Moreover, notice that the features datasets F.6, F.7, and F.8 are the ones that, in general, show a better behaviour with all the models. Therefore, the similarity factor (11), number of terms (13), and number of non-adequate terms (14) features can be considered the most significant ones.

6. Conclusions

The current state of the Web of Data with so many different datasets of heterogeneous nature makes difficult to the users to query them in order to exploit the vast amount of data contained. Different proposals are appearing to overcome that limitation. In this paper we have detailed the features of a framework that allows end users to obtain results from different datasets expressing the query using only the vocabulary which the users are more familiar with, and informs them about the quality of the answer. Moreover, this framework serves technical users as a tool for establishing query rewriting benchmarks.

The framework has been embodied with a selected set of rules, similarity measures, and quality estimation model composed of similarity factor function and F1 score predictive model. Moreover, the framework has been validated in a real scenario and the results obtained are promising, and they could be improved considering smarter rewriting rules and better shaped similarity measures.

7. Acknowledgements

This work is supported by the TIN2013-46238-C4-1-R project and the ELKARTEK program (BID3ABI project).

References

[1] Mario Arias, Javier D Fernández, Miguel A Martínez-Prieto, and Pablo de la Fuente. An empirical study of real-world sparql queries. *Proc. 1st International Workshop on Usage Analysis and the Web of Data, USEWOD*, 2011.

[2] Michelle Cheatham and Pascal Hitzler. The properties of property alignment. In *Proceedings of the 9th International Conference on Ontology Matching-Volume 1317*, pages 13–24. CEUR-WS. org, 2014.

[3] Gianluca Correndo, Manuel Salvadores, Ian Millard, Hugh Glaser, and Nigel Shadbolt. Sparql query rewriting for implementing data integration over linked data. In *Proceedings of the 2010 EDBT/ICDT Workshops*, EDBT '10, pages 4:1–4:11, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-990-9. . URL `http://doi.acm.org/10.1145/1754239.1754244`.

[4] Roberto De Virgilio, Antonio Maccioni, and Riccardo Torlone. A similarity measure for approximate querying over rdf data. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, pages 205–213. ACM, 2013.

[5] Renata Dividino and Gerd Gröner. Which of the following sparql queries are similar? why? In *Proceedings of the First International Conference on Linked Data for Information Extraction - Volume 1057*, LD4IE'13, pages 2–13, Aachen, Germany, Germany, 2013. CEUR-WS.org. URL `http://dl.acm.org/citation.cfm?id=2874472.2874474`.

[6] Shady Elbassuoni, Maya Ramanath, and Gerhard Weikum. Query relaxation for entity-relationship search. In *Proceedings of the 8th Extended Semantic Web Conference on The Semanic Web: Research and Applications*, pages 62–76. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-21063-1. URL `http://dl.acm.org/citation.cfm?id=2017936.2017942`.

[7] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2nd edition, 2013.

[8] Jérôme Euzenat, François Scharffe, and Antoine Zimmermann. D2. 2.10: Expressive alignment language and implementation. *Knowledge Web project report, KWEB/2004/D2.2.10/1.0.*, 2007.

[9] Zong Woo Geem, Joong Hoon Kim, and GV Loganathan. A new heuristic optimization algorithm: harmony search. *Simulation*, 76(2):60–68, 2001.

[10] Peter Haase, Tobias Mathäß, and Michael Ziller. An evaluation of approaches to federated query processing over linked data. In *Proceedings of the 6th International Conference on Semantic Systems*, I-SEMANTICS '10, pages 5:1–5:9, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0014-8. . URL `http://doi.acm.org/10.1145/1839707.1839713`.

[11] Olaf Hartig, Christian Bizer, and Johann-Christoph Freytag. Executing sparql queries over the web of linked data. In *International Semantic Web Conference*, pages 293–309. Springer, 2009.

[12] Aidan Hogan, Marc Mellotte, Gavin Powell, and Dafni Stampouli. Towards fuzzy query-relaxation for rdf. In *ESWC*, volume 7295 of *Lecture Notes in Computer Science*, pages 687–702. Springer, 2012. ISBN 978-3-642-30283-1. URL `http://dblp.uni-trier.de/db/conf/esws/eswc2012.html#HoganMPS12`.

[13] Hai Huang, Chengfei Liu, and Xiaofang Zhou. Computing relaxed answers on rdf databases. In *Web Information Systems Engineering - WISE 2008*, volume 5175 of *Lecture Notes in Computer Science*, pages 163–175. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-85480-7.

[14] Hai Huang, Chengfei Liu, and Xiaofang Zhou. Approximating query answering on rdf databases. *World Wide Web*, 15(1):89–

114, 2012. ISSN 1386-145X. . URL `http://dx.doi.org/10.1007/s11280-011-0131-7`.

[15] Carlos Hurtado, Alexandra Poulovassilis, and Peter Wood. Query relaxation in rdf. *Journal on Data Semantics X*, pages 31–61, 2008. . URL `http://dx.doi.org/10.1007/978-3-540-77688-8_2`.

[16] Mayank Kejriwal and Daniel P. Miranker. An unsupervised instance matcher for schema-free {RDF} data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 35, Part 2:102 – 123, 2015. ISSN 1570-8268. . URL `http://www.sciencedirect.com/science/article/pii/S1570826815000621`. Machine Learning and Data Mining for the Semantic Web (MLDMSW).

[17] Juanzi Li, Jie Tang, Yi Li, and Qiong Luo. Rimom: A dynamic multistrategy ontology alignment framework. *IEEE Transactions on Knowledge and Data Engineering*, 21(8):1218–1232, 2009.

[18] Konstantinos Makris, Nikos Bikakis, Nektarios Gioldasis, and Stavros Christodoulakis. Sparql-rw: transparent query access over mapped rdf data sources. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 610–613. ACM, 2012.

[19] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[20] Alexandra Poulovassilis, Petra Selmer, and Peter T Wood. Approximation and relaxation of semantic web path queries. *Web Semantics: Science, Services and Agents on the World Wide Web*, 40:1–21, 2016.

[21] François Scharffe, Alfio Ferrara, and Andriy Nikolov. Data linking for the semantic web. *International Journal on Semantic Web and Information Systems*, 7(3):46–76, 2011.

[22] Michael Schmidt, Olaf Görlitz, Peter Haase, Günter Ladwig, Andreas Schwarte, and Thanh Tran. Fedbench: A benchmark suite for federated semantic data query processing. In *The Semantic Web–ISWC 2011*, pages 585–600. Springer, 2011.

[23] Ana I. Torre-Bastida, Jesús Bermúdez, and Arantza Illarramendi. Query approximation in the case of incompletely aligned datasets. *Actas de las XX Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2015)*, 2015.

[24] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[25] Jiwei Zhong, Haiping Zhu, Jianming Li, and Yong Yu. Conceptual graph matching for semantic search. In *International Conference on Conceptual Structures*, pages 92–106. Springer, 2002.