What should I link to? Identifying relevant sources and classes for data linking

Andriy Nikolov, Mathieu d'Aquin, Enrico Motta

Knowledge Media Institute, The Open University, Milton Keynes, UK
{a.nikolov, m.daquin, e.motta}@open.ac.uk

Abstract. With more data repositories constantly being published on the Web, choosing appropriate data sources to interlink with newly published datasets becomes a non-trivial problem. It is necessary to choose both the repositories to link to and the relevant subsets of these repositories, which contain potentially matching individuals. In order to do this, detailed information about the content and structure of semantic repositories is often required. However, retrieving and processing such information for a potentially large number of datasets is practically unfeasible. In this paper, we propose an approach which utilises an existing semantic web index in order to identify potentially relevant datasets for interlinking and rank them. Furthermore, we adapt instance-based ontology schema matching to extract relevant subsets of selected data source and, in this way, pre-configure data linking tools.

1 Introduction

The principles of Linked Data[1] recommend data publishers to reuse exiting URIs for their entities, where possible, or to provide links to them. In this way, more information can be obtained by following the links. In order to achieve that, data publishers face two non-trivial problems. First, they must be able to find existing data sources which can be reused or linked to. For this, they must be aware of the content of existing repositories describing relevant domains and be able to assess their suitability. Second, they must configure and run data linking tools which would discover mappings between individuals in their dataset and the chosen external ones.

With the growing number of repositories published based on the Linked Data principles, identifying relevant datasets and resources can become problematic. As a result, data publishers usually only link their datasets to the popular repositories (such as DBPedia[2] and Geonames[3]). This may not always be the optimal solution in some cases, for example:

– If the data domain is highly specialised and not covered by popular repositories in sufficient details.

[1] `http://www.w3.org/DesignIssues/LinkedData`
[2] `http://dbpedia.org`
[3] `http://www.geonames.org/`

– If different parts of the dataset are covered by several external repositories: e.g., when a repository contains references to scientific publications both on computer science (described by DBLP[4]) and medicine (described by PubMed[5]).

To support assessment of different sources, catalogs of Linked Data repositories are maintained (e.g., in CKAN[6]), and meta-level descriptors of repositories are provided using the VoiD vocabulary[7]. However, these sources can still be insufficient as they do not take into account the distribution of instances in repositories. For example, several repositories contain information about academic researchers, however, they use different criteria to include individuals: e.g., DBPedia only mentions the most famous ones, DBLP only includes Computer Science researchers, and RAE[8] deals with researchers working in UK institutions. In order to be able to choose the most appropriate repositories to link to, one must have access to complete instance-level data stored in them. Obtaining these data directly from the data sources and analysing them is often not feasible due to the size of datasets which need to be downloaded.

Once the dataset is chosen, the second challenge is to configure the data linking tool which would discover actual links between individuals in two repositories. The configuration typically includes choosing the selection criterion for determining potential matching candidates and the matching function, which would determine whether a pair of matching candidates actually represent the same entity. Both choices heavily depend on the structure of data in the external data repository.

In this paper we describe an approach which helps to solve these tasks without the need to process complete external datasets. The approach involves two methods, which we consider our contribution:

– Identifying and ranking relevant candidate data repositories for linking. To achieve this, the method utilises keyword-based search over existing an semantic web index, integrates search results, and analyses them.
– Identifying relevant classes containing potentially matching individuals in chosen external sources. To this end, the method adapts and extends instance-based ontology matching techniques. We define the task of finding the best matching class in an external dataset and evaluate the suitability of different instance-based similarity metrics to the task.

The rest of the paper is organized as follows. Section 2 outlines the use case which provided the main motivation for this work. Section 3 describes the method for selecting and ranking the data sources. Section 4 focuses on application of instance-based ontology matching techniques in order to determine a relevant

[4] `http://dblp.l3s.de/`
[5] `http://www.ncbi.nlm.nih.gov/pubmed/`
[6] `http://ckan.net/` see `http://ckan.net/group/lodcloud`
[7] `http://semanticweb.org/wiki/VoiD`
[8] `http://rae2001.rkbexplorer.com/`

subset of the chosen data source. Section 5 discusses the results of the experiments we performed to test our algorithms. Finally, section 7 concludes the paper.

2 Motivation

The problem of determining a set of relevant repositories is a generic one and can occur in different contexts. Our work was primarily motivated by two use cases: the SmartProducts project and development of the *data.open.ac.uk* repository.

2.1 Scenarios and requirements

One of the tasks within the SmartProducts project[9] involves reusing the data from external semantic repositories to build knowledge bases for smart consumer devices: e.g., to extend the core domain knowledge base of food recipes for a smart kitchen with nutritional data, alternative recipes, health profiles of food products, etc. In order to extend the core domain knowledge base, the developer has to be able to find relevant repositories on the Web of Data and interlink them with this core knowledge base [2].

In another scenario, the *data.open.ac.uk* repository[10] aims at publishing various data related to the activities of The Open University (OU)[11] according to Linked Data principles. These datasets include, among others, the publications originated by OU researchers, courses provided by the university, etc. Many entities referenced in these datasets are also mentioned in other public repositories. Thus, in order to facilitate data integration, it makes sense to create links from instances used in the *data.open.ac.uk* datasets to external semantic data stores. Given the range of categories to which data instances belong, it is difficult to select a single external source to link to: e.g., publication venues can be linked to different subsets of RKBExplorer, DBLP, PubMed, DBPedia, or Freebase[12]. Moreover, the repository is constantly extended with more instance data for existing topics (e.g., as more research output is published with time) as well as with more topics (as more internal datasets are released online). Selecting relevant sources for linking and selecting specific individuals to link to within these sources becomes a time-consuming procedure, which needs to be automated as much as possible.

There are several factors which can guide the selection of the repository for linking, in particular:

- *Degree of coverage.* In order to maximise the possibility to reuse external descriptions, the sources which contains more references to the entities stored in the newly published repository are preferable.

[9] `http://www.smartproducts-project.eu`
[10] `http://data.open.ac.uk`
[11] `http://www.open.ac.uk`
[12] `http://www.freebase.com/`

– *Additional information provided by the source.* When selecting a source to link to, it is important to take into account how much additional information about entities is provided by each external source: i.e., what properties and relations are used to describe these entities.
– *Popularity of the source.* Linking to URIs defined in a popular data source or reusing them makes it easier for external developers to find the published data and use them.

Among these factors, only the degree of coverage heavily relies on instance-level data stored in external repositories. The level of detail of instance descriptions can be obtained from the domain ontology used by the external dataset and, possibly, a few example instances, while the popularity of the source can be estimated based on VoiD linkset descriptors. Therefore, when designing our algorithm, we primarily focused on estimating the degree of coverage between the internal dataset prepared for publishing and potentially relevant external datasets.

2.2 Overview of the approach

The task of finding relevant repositories assumes that there is a dataset to be published $D_s = \{O_s, I_s\}$ containing a set of individuals I_s structured using the ontology O_s. We will refer to this dataset as the *source dataset*. Each individual belongs to at least one class c_λ defined in O_s: $I = \{i_j | c_\lambda(i_j), c_\lambda \in O_s\}$. On the Web there is a set of Linked Data repositories $\{D_1, \ldots, D_n\}$ such that $D_j = \{O_j, I_j\}$. There is a subset of these repositories $\{D_1, \ldots, D_m\}$ which overlap with D_s, i.e., $\forall(j \leq m)\exists(I_j^O \subseteq I_j) : I_j^O = \{i_k | equiv(i_k, i_s), i_j \in I_j, i_s \in I_s\}$, where *equiv* denotes the relation of equivalence between individuals. The meaning of the equivalence relation here depends on the identity criterion chosen by the data publisher: e.g., *owl:sameAs* links or direct reuse of URIs assume that URIs must be strictly interchangeable (see [4] for the analysis of different types of identity). The goal is to identify the subset of relevant repositories $\{D_1, \ldots, D_m\}$ and to rank them according to the degree of coverage $|I_j^O|/|I_s|$. Given that the publisher may want to select different repositories to link for different categories of instances in D_s, then for each class $c_\lambda \in O_s$ a separate ranking should be produced based on the degree of coverage for instances of this class $|I_{j\lambda}^O|$, where $I_{j\lambda}^O = \{i_k | equiv(i_k, i_s), i_s \in I_s, c_\lambda(i_s)\} \subseteq I_j^O$.

Since the actual discovery of links is usually performed by an automated tool (such as Silk [14] or KnoFuss [10]), another important task is to restrict the search space for this tool by identifying in each dataset D_j a set of relevant classes c_{jk} which contain potentially overlapping individuals with c_λ. Then the tool can be configured to select only individuals of these classes as candidates for linking. The main obstacle with these tasks is the need to identify the overlapping subset of instances $|I_j^O|$ from each external dataset. Downloading whole datasets or applying data linking tools to their complete sets of instances is often unfeasible due to their size and required computational time, network load, and local disk space.

In order to minimize the amount of data from external repositories which must be processed locally, we adopted a two-stage approach. At the first stage, a semantic web index which supports keyword-based search for data instances is utilised to propose and rank relevant sources as well as potentially relevant classes. This solution, which extends the earlier version of the algorithm presented in [8], is described in section 3. Once a source is selected, the second stage involves finding the relevant classes which would facilitate the data linking process. At this stage, partial information is retrieved from the selected source, in particular, labels of instances of the candidate classes (see section 4).

3 Selecting relevant data sources using keyword search services

We assume that a semantic keyword search service takes as its input a set of keywords $K = \{k_1, \ldots, k_i\}$. As output, it returns a set of potentially relevant individuals which may belong to different repositories: $I^{res} = I_1^{res} \cup I_2^{res} \cup \ldots \cup I_m^{res}$, where $I_j^{res} \subseteq I_j$. For returned individuals $i_{jk} \in I_j^{res}$, their types $\{c_{jk\lambda} | c_{jk\lambda}(i_{jk})\}$ are also available in the search results. An example of the search service which satisfies this assumption is Sig.ma [12], which uses Sindice as its search index.

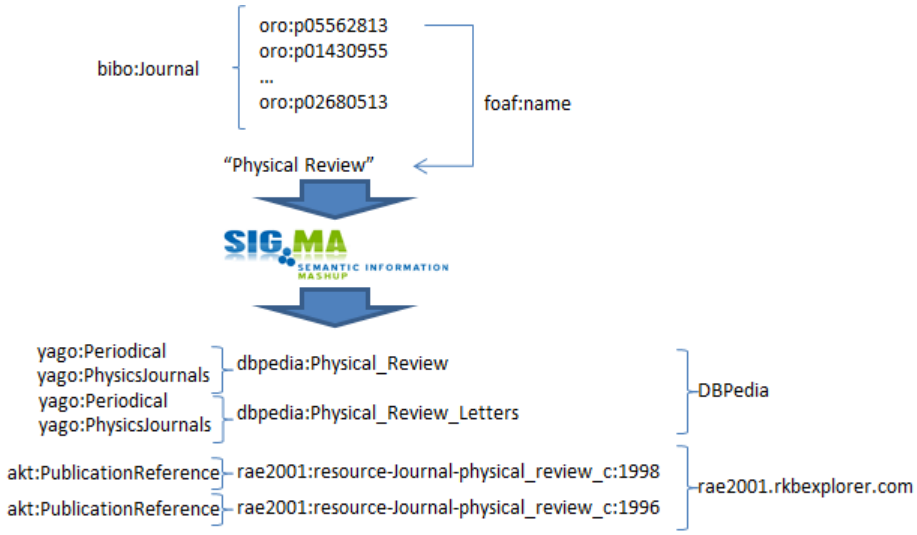

Fig. 1. Keyword-based search for relevant individuals.

3.1 Finding potentially relevant sources

In order to find potentially relevant individuals from the source dataset D_s, we query the search service using the labels of individuals (values of *rdfs:label*,

foaf:name, *dc:title*, etc.) as keywords. Then, these query results are aggregated to estimate the degree of coverage of different data sources (Fig. 1). The procedure consists of the following steps:

1. Randomly selecting a subset of individuals I_s^* from D_s belonging to a class c_s. This is done in order to reduce the number of queries to the search service in case where the complete extension set of individuals is too large. On the other hand, the subset must be large enough to produce reliable ranking of sources.
2. Querying the search service (Sig.ma) for labels of each individual in the selected subset. The results of each search are returned as an RDF document, which includes the references to individuals, their sources, and the classes they belong to.
3. Aggregation of the search results. RDF documents returned by Sig.ma are loaded into a common repository, and the individuals i_{jk} are grouped according to their sources D_j.
4. Data sources are ranked according to the number of their individuals returned by the search service $|\{i_{jk}|i_{jk} \in D_j\}|$.

In our approach we assume that the relevance function used by the search service to select query answers serves as an approximation of the identity function $equiv()$. In the general case, this is in not true due to ambiguity of labels and the fact that search services may not always achieve 100% precision. Taking a sufficiently large subset of individuals to search makes it possible to reduce the impact of 'false positives' returned by the search engine.

After applying these steps to our test scenarios (see section 5), we found that the rankings obtained using this procedure are still likely to be imprecise for two main reasons:

- Inclusion of irrelevant sources. For individuals belonging to classes with highly ambiguous labels, many 'false positives' in the set of answers can result in irrelevant repositories achieving high ranking positions. For instance, when searching for specific subcategories of people, any source mentioning sufficiently large number of people would be considered relevant: e.g., Twitter and DBLP were highly ranked when searching for music contributors.
- Inclusion of irrelevant classes. Resulting sets often contained classes which would not allow selecting appropriate candidate individuals by a matching tool. Sometimes a generic superclass was ranked higher than the correct class: e.g., *dbpedia:Person* was ranked higher than a more relevant *dbpedia:MusicalArtist*. In other cases, completely irrelevant classes were included: e.g., for scientific journals the class *akt:Publication-Reference* describing specific volumes of journals was ranked higher than *akt:Journal*.

In order to overcome these issues, our approach includes the second stage: filtering of search results using ontology matching techniques.

3.2 Using ontology matching techniques to filter out irrelevant results

In order to filter out irrelevant search results, our approach can utilise mappings between classes provided by existing schema matching tools (Fig. 2). In our

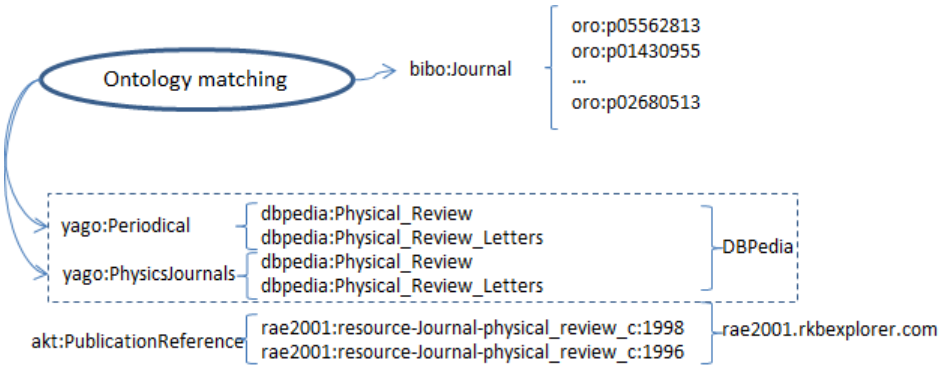

Fig. 2. Using ontology matching to refine search results.

experiments we utilised ontology mappings produced by two algorithms:

- CIDER [3] which takes as input two ontologies in RDF format and two URIs defining ontological terms from these ontologies and produces as output the similarity score between these terms. CIDER utilises evidence defined at the level of ontological schema: string similarity between class labels, semantic relations defined in WordNet and positions of classes in class hierarchies.
- Instance-based matching algorithm described in [9], which generated schema mappings between classes on the Web of Data based on their overlapping sets of instances. Overlapping sets of instances were inferred based on existing *owl:sameAs* relations between them published in the Billion Triple Challenge 2009 (BTC) dataset[13]. Resulting mappings represent subsumption relations of the form $c_A \sqsubseteq c_B$, where c_A and c_B belong to different ontologies.

As the first step of the filtering procedure, CIDER is applied to measure similarity between the class c_s in D_s, for which overlapping sources have to be found, and each of the classes $c_{jk\lambda}$ appearing in the aggregated search results. Then, a threshold is applied to filter out classes with low similarity scores. Remaining classes from the search results constitute the set of 'confirmed' classes $C_{confirmed}$. At the next stage, this set of 'confirmed' classes is enriched using the mappings obtained using instance-based matching. For each class $c_i \in C_{confirmed}$, all mappings from the BTC-based set where $c_A \sqsubseteq c_i$ are selected, and all c_A are added into $C_{confirmed}$. After the filtering stage, the datasets for which there is at least

[13] `http://vmlion25.deri.ie/`

one 'confirmed' class are moved in the ranking above those for which no classes were confirmed.

In our tests described in section 5, the filtering stage led to improved precision in the resulting ranking of data sources. However, the approach was found insufficient to deal with the actual task of finding relevant classes in target repositories. While the main problem with unfiltered results was the choice of too generic classes, the filtering procedure left out many relevant classes or chose too specific classes in the hierarchy. Because of this, the special method was implemented to identify the best-matching classes in the ontology of a given dataset.

4 Identifying relevant classes in the dataset

Identifying a relevant repository for interlinking represents only the first stage of the process. In order to configure the data linking method and minimize the possible errors, it is important to select the relevant subset of instance containing potentially coreferent individuals. Selecting too broad subset can substantially increase the computational time required to compare irrelevant individuals and can also lead to many spurious mappings, thus reducing precision. Selecting too small subset, on the other hand, can lead to missing mappings and reduced recall. Given that the instances in semantic repositories are organised using ontological class hierarchies, selecting a relevant subset of data for interlinking requires selecting the best fitting class in the target ontology.

Definition 1*: Let I_s represent a set of individuals from the source repository D_s. A subset of these individuals I_s^O has matching individuals in the target repository D_t: $\forall i_s \in I_s^O \exists i_t \in I_t^O : i_s \equiv i_t$. Then the best fitting class for I_s is such a class $c_t^{fit} \in D_t$ that it contains all individuals from I_t^O and there is no subclass $c_x \sqsubseteq c_t^{fit}$ that $\forall i_t \in I_t^O : i_t \in c_x$.*

Assuming that all instances $i_s \in I_s$ belong to the same class c_s, the task of choosing a best-fitting class represents a special case of the ontology matching problem. However, it has several specific features [11]:

- It is possible that not all instances of the source class have matching counterparts in the target repository. Thus, the goal is to find a fuzzy 'overlap' relation between classes rather than strict logical equivalence or subsumption.
- Class definitions in the ontology can be insufficient to capture the intended meaning: e.g., the class *Actor* in *LinkedMDB* refers to any person participating in a movie, while *Actor* in *DBPedia* refers to professional actors (both film and stage).

Because of this, instance-based ontology matching techniques, which determine relations between classes based on the overlap between their instance sets, appear especially suitable for the task. However, these techniques cannot be directly reused: in the absence of mappings between individuals in two repositories, it is impossible to determine the overlap between their instance sets. Thus, applying

instance-based ontology matching to the task of determining the most relevant class in the hierarchy requires dealing with two challenges:

- Approximating the power of the overlapping set of instances for two classes in the absence of actual instance mappings.
- Selecting a suitable set-based similarity measure, which can determine the degree of relevance of a particular class c_j for a set of instances I_s.

In order to estimate the power of the overlap relation between two classes, we use the same evidence as for the source ranking: keywords extracted from instance labels. The algorithm takes as its input the selected subset of individuals from the source dataset as well as the output of the source selection algorithm: the target repository D_t and the initial class c_t^{top}. As our tests have shown, the class c_t^{top} returned by the source selection procedure is usually too generic. The goal of the algorithm is to find the 'best-fitting' subclass $c_t^{fit} \sqsubseteq c_t^{top}$. The procedure consists of the following steps:

1. Create profiles of all instances $i_{tj} \in c_t^{top}$. A profile $P(i_{tj})$ of the instance i_{tj} includes all keywords extracted from the label of i_{tj}.
2. Randomly select a subset of individuals I_s^* from I_s.
3. For each individual $i_s \in I_s^*$, find the 'best matching' individual i_{tj}^+ using the cosine similarity between individual profiles: $i_{tj}^+ = argmax(cosim(P(i_s), P(i_{tj})))$. For each class c_{ti} such that $i_{tj}^+ \in c_{ti}$, increase the score $s(c_i)$.
4. For each subclass $c_{ti} \sqsubseteq c_t^{top}$, its score $s(c_{ti})$ serves as an estimation of the overlap $|c_s \cap c_{ti}|$. Based on this overlap estimation, the similarity between classes is calculated $sim_i(|c_s \cap c_{ti}|) \approx sim_i(s(c_{ti}))$. The class c_{ti} with the highest similarity degree is assigned as c_t^{fit}.

Obtaining the best matching target individual is implemented using a standard keyword-based search mechanism using an in-memory Lucene index. This procedure cannot be used as a replacement for the actual instance matching tools due to its low accuracy, but, given a sufficient sample size, it can approximate instance equivalence in order to produce schema-level links. At this stage, the sample from the first step of the algorithm (search for potentially relevant sources) can be reused.

In order to measure the actual similarity between two classes with overlapping sets of instances, several metrics have been used, in particular:

- The *Jaccard index* is defined as $JC(I_1, I_2) = \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|}$. In [5] a modified version was proposed to give advantage to classes with large number of instances. This *corrected Jaccard index* is defined as $JC_{corr} = \frac{\sqrt{|I_1 \cap I_2| \times (|I_1 \cap I_2| - 0.8)}}{|I_1 \cup I_2|}$.
- *Overlap coefficient* is another set similarity metrics defined as $Overlap(I_1, I_2) = \frac{|I_1 \cap I_2|}{min(|I_1|, |I_2|)}$. It can reduce the impact of situations in which classes of one dataset contain substantially less individuals than in the other one. However, it is often incapable of ranking several alternative mappings with the same size of the overlap.

– *Pointwise Mutual Information* determines the reduction of uncertainty provided by the assignment of an instance to one class to the assignment to the other: $PMI(I_1, I_2) = log_2 \frac{|I_1 \cap I_2| \times N}{|I_1| \times |I_2|}$.
– Log likelihood ratio represents a statistical test used to compare the fit of two hypotheses. The null hypothesis states that the probability $p(i \in I_1)$ that an instance belongs to I_1 does not depend on whether it already belongs to I_2, i.e., $p_0 = p(i \in I_1 | i \in I_2) = p(i \in I_1 | \neg i \in I_2) = \frac{|I_1|}{N}$, where N is a total number of instances in both datasets. The alternative hypothesis states that $p_1 = p(i \in I_1 | i \in I_2) = \frac{|I_1 \cap I_2|}{|I_2|}$ and $p_2 = p(i \in I_1 | \neg i \in I_2) = \frac{|I_1| - |I_1 \cap I_2|}{N - |I_2|}$. The log likelihood ratio is defined as $-2(logL(p_0, k_1, n_1) + logL(p_0, k_2, n_2) - logL(p_1, k_1, n_1) - logL(p_2, k_2, n_2))$, where $logL(p, k, n) = klnp + (n-k)ln(1-p)$, $k_1 = |I_1 \cap I_2|$, $k_2 = |I_1| - |I_1 \cap I_2|$, $n_1 = |I_2|$, and $n_2 = N - |I_2|$.
– *Information Gain* measures the reduction of entropy of assigning an instance to one set, if it has already been assigned to another set. $IG = e_1 - e_2$, where $e_1 = -\frac{|I_2|}{N} log_2 \frac{|I_2|}{N}$ and $e_2 = -\frac{|I_1 \cap I_2|}{|I_1|} log_2 \frac{|I_1 \cap I_2|}{|I_1|}$.

An empirical study [5] found the Jaccard index to be the most suitable similarity measure for instance-based ontology matching. However, this study was primarily aimed at identifying equivalence mappings, and the experiments were performed with the ontologies which actually had overlapping sets of instances. Because of this, we decided to perform experiments to evaluate the suitability of different similarity metrics for determining the best-fitting classes.

5 Experiments

We performed two sets of experiments. First, we tested the dataset selection algorithm in three different scenarios (section 5.1). Second, we performed experiments with the algorithm identifying best matching classes in order to choose the instance-based similarity measure best suited to the task.

5.1 Dataset search

In our tests, we have applied the approach described in section 3 to the following datasets:

– ORO journals. A set of 3110 journals mentioned in the ORO repository constituting a part of *data.open.ac.uk*. Each individual belongs to the class *bibo:Journal*[14].
– LinkedMDB films. A subset of 400 randomly selected instances of the class *movie:film*[15] representing movies in the LinkedMDB repository.
– LinkedMDB music contributors. A subset of 400 randomly selected instances of the class *movie:music_contributor* representing music contributors for films in the LinkedMDB repository.

[14] `http://purl.org/ontology/bibo/Journal`
[15] `http://data.linkedmdb.org/movie/film`

For each individual in these sets, we queried Sig.ma using their labels as keywords. The search results containing potentially relevant instances were aggregated, and individuals were grouped by data source and ontological class. These grouped results were used to produce the ranking of sources as described in section 3.1. Among the top-10 ranked data sources, we counted the number of actually relevant ones. Then, we applied the filtering mechanism using ontology schema matching results and checked the relevance of remaining sources. The results we obtained are presented in Table 1: for each dataset it shows the list of top ranked sources as well as our judgement whether these sources were actually relevant (column "+/-"). In the table, "(RKB)" denotes the datasets from RKBExplorer and "open EAN" corresponds to *openean.kaufkauf.net*. For both LinkedMDB datasets, we did not consider the LinkedMDB repository itself when it was returned in the search results. As we can see from the results, the initial search-

Table 1. Test results: ranking of data sources

Dataset	Before filtering		After filtering	
	Top-ranked	+/-	Top-ranked	+/-
Journals	rae2001(RKB)	+	rae2001(RKB)	+
	dotac(RKB)	+	DBPedia	+
	DBPedia	+	dblp.l3s.de	+
	oai(RKB)	+	Freebase	+
	dblp.l3s.de	+	DBLP(RKB)	+
	wordnet(RKB)	-	eprints(RKB)	+
	www.bibsonomy.org	-		
	eprints(RKB)	+		
	Freebase	+		
	www.examiner.com	-		
Films	DBPedia	+	DBPedia	+
	open EAN	+	Freebase	+
	bestbuy.com	+		
	Freebase	+		
	www.answers.com	-		
	bitmunk.com	-		
	wordnet	-		
	www.examiner.com	-		
	it.bestshopping.com	+		
	www.songkick.com	-		
Musicians	DBPedia	+	Freebase	+
	www.realpageslive.com	-	DBPedia	+
	twitter.com	-		
	BBC	+		
	www.songkick.com	+		
	Freebase	-		
	Open EAN	+		
	LinkedIn	-		
	dblp.l3s.de	-		
	Yahoo!Movies	+		

based ranking managed to discover relevant datasets for the sets of individuals in question. Top-ranked sources in the *Journals* and *Films* categories contained relevant individuals which could be linked to the individuals in D_s, and their sets of individuals are to a large degree overlapping. For music contributors, the proportion of irrelevant sources was substantially larger due to higher ambiguity of human names. The filtering stage in all cases resulted in improving the ranking

precision: only relevant sources were confirmed. However, if we look at the rank-

Table 2. Test results: ranking of ontological classes.

~~Dataset~~	Before filtering	After filtering	Best-fitting classes
Journals	*akt:Publication-Reference*	*akt:Journal*	*dc:BibliographicResource*
	dc:BibliographicResource	*yago:Periodical*	*akt:Publication-Reference*
	foaf:Document	*swrc:Journal*	*akt:Journal*
	swrc:Publication	*dbpedia:Work*	*yago:Periodical*
	vcard:VCard	*freebase:book.periodical*	*dbpedia:Work*
	yago:Periodical		*freebase:book.periodical*
	geo:SpatialThing		
	wn:Word		
	rss:item		
	swap:SocialEntity		
Films	*dbpedia:Work*	*dbpedia:Film*	*dbpedia:Film*
	goodrelations: ProductOrServiceModel	*yago:Movie*	*goodrelations: ProductOrServiceModel*
	yago:Movie	*freebase:film.film*	*yago:Movie*
	icalendar:Vevent		*freebase:film.film*
	foaf:Person		*searchmonkey:Product*
	vcard:VCard		
	searchmonkey:Product		
	skos:Concept		
	geo:SpatialThing		
	freebase:common.topic		
Musicians	*vcard:VCard*	*freebase:film. music_contributor*	*freebase:film. music_contributor*
	geo:SpatialThing	*yago:American TelevisionComposers*	*mo:MusicArtist*
	swap:Person		*dbpedia:Artist*
	foaf:Person		*yago:Composer*
	dc:Agent		
	mo:MusicArtist		
	icalendar:vcalendar		
	dbpedia:Person		
	goodrelations:ProductOrService		
	frbr:ResponsibleEntity		

ing of ontological classes (Table 2), we can see that correctly identifying classes presents a number of issues. The table shows the highest ranking classes returned after each stage of the algorithm (only one highest-ranking class from each ontology is shown). Top-ranked classes produced from the search results usually represent high-level concepts and correspond to superclasses of the original class: e.g., *foaf:Document* or *dc:BibliographicResource* for journals, *dbpedia:Work* for movies, and *foaf:Person* for musicians. The filtering stage largely removed these problems so that only classes with a stronger degree of semantic similarity were confirmed. However, it also reduced the recall in cases where a directly corresponding class was not present in the external ontology: e.g., individuals from *dotac.rkbexplorer.com* and *oai.rkbexplorer.com*, which only used the generic class *dc:BibliographicResource* were not considered as relevant sources for linking journals. Similarly, many relevant classes were filtered out because they were not considered as exact matches or subclasses of the class *movie:music_contributor* (e.g., *mo:MusicArtist* and *dbpedia:MusicalArtist*). In other cases, the algorithm selected too specific class, such as *yago:AmericanTelevisionComposers*. Apply-

ing the best-fitting class selection procedure in these cases (column 4) provided more adequate results.

5.2 Finding the best-fitting class

In order to evaluate different set similarity metrics for the best-fitting class, we needed a set of multiple test cases. Each test case required the availability of gold standard mappings between instances as well as ontologies with detailed class hierarchies. To generate sufficient number of such test cases, we have chosen two large-scale datasets which has already been linked: DBPedia and Freebase. Pairs of classes for tests were selected from the YAGO ontology and the Freebase schema. We selected such pairs of classes $(c_y; c_f)$ from YAGO and Freebase respectively that:

- There is a set of *owl:sameAs* mappings $M_i = \{(i_y, i_f)\}$ such that $\forall i_y, i_f : i_y \in c_y, i_f \in c_f$.
- There is a pair of classes $(c_y^{top}; c_f^{top})$ such that $c_y \sqsubseteq c_y^{top}$, $c_f \sqsubseteq c_f^{top}$, and $c_y^{top} \equiv c_f^{top}$.
- There is no such class c_x such that $|c_x| < |c_y|$ and, given $M_i = \{(i_y, i_f)\}$, all i_y would belong to c_x. The same holds for c_f and i_f, respectively.

We selected medium-size classes from Freebase and DBPedia (having between 400 and 20000 individuals) with at least 400 mappings between them, coming from two different domains: people and organisations. After eliminating classes which did not satisfy the criteria or semantically irrelevant ones, the test set contained 111 pairs of classes. For each test, we randomly selected n individuals for which *owl:sameAs* mappings existed, and used them as I_s^{ast}. For these individuals, we ran the procedure described in section 4 using different sim_i and the sample size n. If the procedure returned the actual target class as the best fitting one, the result was considered correct. The test results are summarised in Table 3 (numbers show the proportion of correctly identified target classes). As can be seen, the log likelihood ratio clearly outperforms other metrics both in terms of absolute performance and robustness. The PMI, IG, and $Over$

Table 3. Test results: finding the best fitting classes.

N	sim_i	$n = 50$	$n = 100$	$n = 200$	n=400
1	Jaccard index, JC	0.25	0.46	0.61	0.74
2	Corrected Jaccard index, JC_{corr}	0.41	0.51	0.65	0.74
3	Log likelihood ratio, $LogL$	**0.93**	**0.96**	**0.97**	**0.98**
4	Pointwise mutual information, PMI	0.12	0.07	0.06	0.05
5	Information gain, IG	0.0	0.0	0.0	0.0
6	Overlap coefficient, $Over$	0.0	0.0	0.0	0.0

measures were found to be unsuitable for the task. While they usually return semantically correct class mappings, they tend to select too specific classes in the hierarchy.

6 Related work

Although both the problem of search in semantic datasets and the task of data interlinking are actively studied in the Semantic Web community, there has been relatively little research dedicated to the task of search for relevant datasets. One recent approach [7] also discusses the problem of integrating a dataset with external semantic resources. As a use case, the authors consider the Google Refine application[16] scenario: enriching data from tabular sources. The authors describe an extension to this application capable of linking these tabular data to external semantic repositories and discuss applicable linking techniques (e.g., SPARQL extension and reuse of Sindice and Silk services). However, their experiments only compare these techniques on the task of linking to pre-defined data sources, and do not focus on the actual search for relevant sources. The OKKAM project[17] took a radical centralised approach, in which a global repository of entities exists and provides lookup services for other datasets to retrieve canonical URIs for their data instances.

To deal with the task of identifying matching classes, instance-based matching techniques are actively researched in the ontology matching community [1] and incorporated in several schema matching tools (e.g., ILIADS [13] and RiMOM [6]). In particular, in [15] the authors use the 'bag of words' approach adapted from the natural language processing: classes are annotated with the sets of string tokens extracted from properties of their instances, and similarity between classes is measured using the cosine similarity. However, this technique loses the information about distribution of words in different instances and is not suitable for estimation of the overlap between instance sets. As mentioned in section 4, the comparative study reported in [5] evaluated the suitability of different similarity metrics, although the focus of their task and their conclusions differ from ours.

7 Conclusion

The Linked Data cloud is constantly growing, and in order to make its use widespread, data owners must be able to publish their datasets without extensive knowledge about the state of the Web of Data or assistance from the research community. Interlinking is an important part of the publishing process and the one which can require substantial exploratory work with external data. Thus, this process has to become straightforward for data publishers and, preferably, require minimal human involvement. A specific feature of this problem is the fact that the amount of necessary information about the Web of Data which is immediately available on the client (data publisher) side is limited, and gathering this information is a time-consuming process for the user. The proposed solution provides the data publisher with a ranked set of potentially relevant data sources and, in addition, a partial configuration of the data linking tool

[16] `http://code.google.com/p/google-refine/`
[17] `http://www.okkam.org`

(classes containing relevant sets of instances). In this way, it can substantially reduce the need to perform exploratory search. One direction of the continuation work, which we are currently pursuing, involves developing algorithms which are able to suggest to the user suitable instance matching algorithms for the data linking tool depending on the task at hand.

Another potentially interesting research direction is related to the development of semantic indexes. Search for relevant data repositories can become a novel interesting use case in addition to the more common search for entities and documents. In order to support it, new types of search services can be valuable: for example, batch search for a large array of resource labels instead of multiple queries for small sets of keywords, which increase number of server requests and overall processing time.

8 Acknowledgements

This research has been partially funded under the EC 7th Framework Programme, in the context of the SmartProducts project (231204).

References

1. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg, 2007.
2. M. Fernandez, Z. Zhang, V. Lopez, V. Uren, and E. Motta. Ontology augmentation: combining semantic web and text resources. In *6th International Conference on Knowledge Capture (K-CAP 2011)*, 2011.
3. J. Gracia and E. Mena. Matching with CIDER: Evaluation report for the OAEI 2008. In *3rd Ontology Matching Workshop (OM'08) at the 7th International Semantic Web Conference (ISWC'08)*, Karlsruhe, Germany, 2008.
4. H. Halpin, P. J. Hayes, J. P. McCusker, D. L. McGuinness, and H. S. Thompson. When owl:sameas isn't the same: An analysis of identity in linked data. In *9th International Semantic Web Conference (ISWC 2010)*, pages 305–320, Shanghai, China, 2010.
5. A. Isaac, L. van der Meij, S. Schlobach, and S. Wang. An empirical study of instance-based ontology matching. In *6th International Semantic Web Conference*, pages 253–266, Busan, Korea, 2007.
6. J. Li, J. Tang, Y. Li, and Q. Luo. RiMOM: A dynamic multistrategy ontology alignment framework. *IEEE Transactions on Knowledge and Data Engineering*, 21(8):1218–1232, 2009.
7. F. Maali, R. Cyganiak, and V. Peristeras. Re-using cool URIs: Entity reconciliation against LOD hubs. In *Workshop on Linked Data on the Web (LDOW 2011), WWW 2011*, Hyderabad, India, 2011.
8. A. Nikolov and M. d'Aquin. Identifying relevant sources for data linking using a semantic web index. In *Workshop on Linked Data on the Web (LDOW 2011), WWW 2011*, Hyderabad, India, 2011.
9. A. Nikolov and E. Motta. Capturing emerging relations between schema ontologies on the web of data. In *Workshop on Consuming Linked Data (COLD 2010), ISWC 2010*, Shanghai, China, 2010.

10. A. Nikolov, V. Uren, E. Motta, and A. de Roeck. Integration of semantically annotated data by the KnoFuss architecture. In *16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, pages 265–274, Acitrezza, Italy, 2008.
11. A. Nikolov, V. Uren, E. Motta, and A. de Roeck. Overcoming schema heterogeneity between linked semantic repositories to improve coreference resolution. In *4th Asian Semantic Web Conference (ASWC 2009)*, pages 332–346, Shanghai, China, 2009.
12. G. Tummarello, R. Cyganiak, M. Catasta, S. Danielczyk, R. Delbru, and S. Decker. Sig.ma: Live views on the Web of Data. *Journal of Web Semantics*, 8(4):355–364, 2010.
13. O. Udrea, L. Getoor, and R. J. Miller. Leveraging data and structure in ontology integration. In *SIGMOD'07*, pages 449–460, Beijing, China, 2007.
14. J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov. Discovering and maintaining links on the Web of Data. In *8th International Semantic Web Conference (ISWC 2009)*, pages 650–665, Washington, DC, USA, 2009.
15. S. Wang, G. Englebienne, and S. Schlobach. Learning concept mappings from instance similarity. In *7th International Semantic Web Conference (ISWC'08)*, pages 339–355, Karlsruhe, Germany, 2008.