Online Relation Alignment for Linked Datasets

Maria Koutraki[1,2](✉), Nicoleta Preda[3], and Dan Vodislav[4]

[1] FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, Karlsruhe, Germany
[2] Institute AIFB, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
maria.koutraki@kit.edu
[3] Universtiy of Paris-Saclay, Versailles, France
nicoleta.preda@uvsq.fr
[4] ETIS CNRS, University of Cergy-Pontoise, Cergy-Pontoise, France
dan.vodislav@u-cergy.fr

Abstract. The large number of linked datasets in the Web, and their diversity in terms of schema representation has led to a fragmented dataset landscape. Querying and addressing information needs that span across disparate datasets requires the alignment of such schemas. Majority of *schema* and *ontology alignment* approaches focus exclusively on class alignment. Yet, relation alignment has not been fully addressed, and existing approaches fall short on addressing the dynamics of datasets and their size.

In this work, we address the problem of *relation alignment* across disparate linked datasets. Our approach focuses on two main aspects. First, *online relation alignment*, where we do not require full access, and sample instead for a minimal subset of the data. Thus, we address the main limitation of existing work on dealing with the large scale of linked datasets, and in cases where the datasets provide only *query access*. Second, we learn supervised machine learning models for which we employ various features or *matchers* that account for the diversity of linked datasets at the instance level. We perform an experimental evaluation on real-world linked datasets, DBpedia, YAGO, and Freebase. The results show superior performance against state-of-the-art approaches in schema matching, with an average relation alignment accuracy of 84%. In addition, we show that relation alignment can be performed efficiently at scale.

1 Introduction

In the recent years, the number of datasets exposed as linked data has grown continuously. Estimates show that there are more than 1000 datasets, with roughly 30 billion facts in the form of triples [1]. The decentralized nature of these datasets and furthermore, the lack of mechanisms and proper documentation for reusing existing schemas has led to a large number of schemas, 650 in LOD [25]. In many cases classes and relations across such schemas are redundant and not aligned (with only 2% of schemas aligned [25]). As a consequence this leads to a disintegrated dataset landscape [9].

E. Blomqvist et al. (Eds.): ESWC 2017, Part I, LNCS 10249, pp. 152–168, 2017.
DOI: 10.1007/978-3-319-58068-5_10

This particular problem has been partly addressed by *ontology alignment* approaches, which almost exclusively have focused on class alignment across disparate ontologies [27]. In contrast to the class alignment, *relation alignment* has not seen such progress. Yet an equally important task, helping address information needs that span across datasets. Through both class and relation alignment we can rewrite and perform federated queries across disparate datasets.

The few existing relation alignment approaches, like the state of the art [28] fail to incorporate two main aspects of linked datasets. First, [28] is a blocking solution that requires several iterations over full datasets and which takes several hours to complete. As shown in [25] full dataset access is not always possible. Many datasets limit to only *query access* and to a few thousands of triples per query. Secondly, it does not take into account the *evolving nature* of datasets. Upon updates, only some of the alignments may require to be modified. Yet, the existing solutions need to be run from scratch on the full datasets, thus making such approaches inefficient.

We propose *SORAL – Supervised Ontology Relation ALignment*, a supervised machine learning approach for *relation alignment* across disparate schemas and datasets. Similar to [17], we work under the assumption that `owl:sameAs` statements exist between entities across datasets [25]. Additionally we operate in an online setting requiring only SPARQL query access to datasets, thus allowing for large scale alignment, independent of dataset size.

Figure 1 shows *SORAL*, consisting of two main steps. First, for a *source relation* and source dataset, we generate *candidate relations* for alignment to a *target dataset*. As candidates we consider relations that have entity instances in common (based on `owl:sameAs` statements). In this step we employ *sampling strategies* to cope with the scale of datasets and ensure the efficiency of *SORAL*. Second, for a relation pair, we compute features from the common entity instances, and information we extract from the relations, e.g. *domain/range* of relations etc. *SORAL* optimizes for *efficiency* by performing the alignment in an online setting, and *accuracy* by providing qualitative relation alignments. To this end, we make the following contributions:

- We propose an approach for finding relation subsumptions and equivalences;
- We perform the relation alignment in an *online* setting that ensures *efficiency*;
- We conducted an extended experimental evaluation using real-world knowledge bases (KBs).

2 Related Work

Scope. Our goal is to discover *subsumption* and *equivalence* relationships between the relations of two KBs exported using SPARQL endpoints. This problem is also known as *ontology alignment*. Ontology alignment includes: *instance alignment*, *class alignment*, and *relation alignment*. To align relations, we rely on existing work on aligning instances [5,6,20]. Hence, we see this effort as complementary to ours.

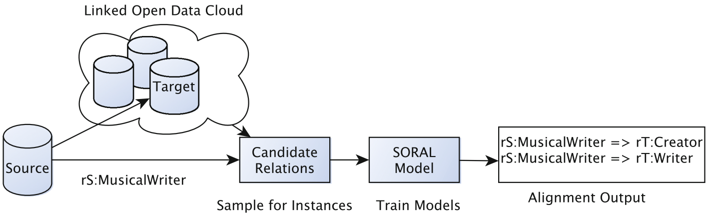

Fig. 1. SORAL approach overview.

Related work in this field is usually categorized in *schema-based* and *instance-based* approaches depending on the data that is used to produce the alignments. The majority of the work focuses on *class alignment* and much less on *relation alignment*, which is the objective of this work.

Schema-Based. Schema matching systems like in [3,8,22,26] align classes by relying on schema constrains, which are often unavailable in LOD. COMA++ [3] expects as input OWL descriptions and ignores data instances. While in [8] the approach relies on user assistance for the alignment process. Other systems like in [26] serve as proxy over existing schema matching tools, thereby providing means on combining the different functionalities of the individual platforms. BLOOMS [15], aligns classes in different KBs where the schema definitions are available. In our vision, the alignment of KBs should happen even in cases where explicit schema information or other database constraints are not available. For that reason we choose an instance-based approach.

Instance-Based: Class Alignment. Many instance-based approaches are proposed for class alignment in ontologies [12,16,31], contrary to our work, that focuses on relation alignment. Movshovitz-Attias et al. [23] infer *subconcept* and *superconcept* hierarchies from a KB. [24] produces equivalence and subsumption relationships between classes. Contrary to the class alignment work, the difference in the case of relations lies on two main aspects. First, classes and relations represent two different constructs in schemas. For classes it is sufficient to simply measure set overlap in terms of entities and other schematic information. In the case of relations we have the domain and range which should map between two relations, and even in cases they match, two distinct relations might have different semantics (e.g. `bornIn` and `presidentOf`).

Instance-Based: Relation Alignment. A large corpus of works focus on the discovery of *equivalence* or *similarity* relationships between relations, with early examples in relational databases [11,21], and Web services [18,19]. In the case of KBs, the schemas are in large numbers and vary heavily. For example, YAGO and DBpedia vary greatly in their representation, where YAGO has 37 object relations, while DBpedia has 688, whereas in terms of entity instances they have a high overlap. Thus, by considering only *equivalence* alignments we cannot map relations that have similar semantics or subsume another. This is due to the fact that many relations map to more than one relation in a target KB (e.g. YAGO to DBpedia). In itself finding *subsumption* relationships is more challenging, yet, it reflects the true nature of this problem. In the following we describe only the approaches that are capable of handling subsumption relationships [28,30,32].

ILIADS [30] and PARIS [28] serve for the dual purpose of aligning instances and schema elements between two KBs. In ILIADS as in [7], the authors acknowledge the difficulty of the relation alignment problem, where systems designed for class and instance alignment perform poorly. In the case of PARIS, it is acknowledged as a state of the art approach by related work [32], thus, we compare against it, and show the advantages of *SORAL* achieving significant improvement over PARIS. Galárraga et al. [13] propose *ROSA*, a relation alignment approach. *ROSA* operates on a similar setting as ours. However, as we show in our experimental evaluation in Sect. 7, relying solely on *association rule mining* measure PCA [14] is insufficient. Finally, the aforementioned works [13,28,30,32] operate on KB snapshots, contrary to our approach, which operates on small data samples that are queried from the respective KBs, thus, tackling one of the major problems of *efficiency* of existing works.

3 Preliminaries

Knowledge Bases. We assume that knowledge bases (KB) are represented in RDFS[1]. A KB consists of a set of triples $\mathcal{K} \subseteq \mathcal{E} \times \mathcal{R} \times (\mathcal{E} \cup \mathcal{L})$, where $\mathcal{E}$ is a set of resources that we refer as entities, $\mathcal{L}$ a set of literals, and $\mathcal{R}$ a set of relations. An *entity* is identified by a URI and represents a real-world object or an abstract concept. A *relation* (or property) is a binary predicate. A *literal* is a string, date or number.

Given a triple (x, r, y) (aka a statement), x is known as subject, r as relation, and y as object. Based on the nature of the objects, we classify relations in two categories: (i) *entity-entity* relations with x and y being entities, and (ii) *entity-literal* relations where y is a literal.

We use $r(x, y)$ to refer to the triple (x, r, y), and the tuple (x, y) as an *instantiation* of r. Without loss of generality, for an entity-entity relation $r \in \mathcal{R}$ we denote its inverse as $r^- \in \mathcal{R}$, and that the triples of r^- are also contained by $\mathcal{K}$ $(\forall r(x, y) \in \mathcal{K} \Rightarrow r^-(y, x) \in \mathcal{K})$.

Classes and Instances. A *class* is an entity that represents a set of objects, e.g., the class of all politicians. An entity that is a member of a class is called an *instance* of that class. The `rdf:type` relation connects an instance to a class. For example, *Barack_Obama* is a member of the class of politicians: `rdf:type`*(Barack_Obama, Politician).*

Equivalence of Instances. Equivalence between two entities is expressed through `owl:sameAs` statements. For example, *yago:US* and *dbpedia:USA* (both referring to the same country), the equivalence of the two instances is expressed as `owl:sameAs`(*yago:US, dbpedia:USA*).

---

[1] https://www.w3.org/RDFS/.

4 Problem Statement

We consider a KB pair (a source $\mathcal{K}_S$ and a target $\mathcal{K}_T$) that can be queried through SPARQL endpoints. Next, we assume that `owl:sameAs` statements exist between entity instances of $\mathcal{K}_S$ and $\mathcal{K}_T$. For two equivalent instances $e_s \equiv e_t$, from $\mathcal{K}_S$ and $\mathcal{K}_T$, respectively, we assume that, $\mathcal{K}_S$ stores `owl:sameAs`(e_s, e_t) and $\mathcal{K}_T$ stores `owl:sameAs`(e_t, e_s).

Definition 1 (Relation Subsumption). *For two relations r_S and r_T, we say that r_S is* subsumed *by r_T, or that r_T* subsumes *r_S and write $r_S \subseteq r_T$ or $r_S \Rightarrow r_T$* iff

$$\forall x, y, r_S(x, y)\, is \text{ true} \rightarrow r_T(x, y)\, is\ also \text{ true}.$$

The notion of subsumption is independent of the relation extensions/name in the two KBs. We can only rely on the facts stored in the KBs to learn the subsumption relationships.

Goal. *For two knowledge bases, a source $\mathcal{K}_S(\mathcal{E}_S, \mathcal{R}_S, \mathcal{E}_S \cup \mathcal{L}_S)$ and a target $\mathcal{K}_T(\mathcal{E}_T, \mathcal{R}_T, \mathcal{E}_T \cup \mathcal{L}_T)$, and a relation $r_S \in \mathcal{R}_S$, find the relations $r_T \in \mathcal{R}_T$ s.t. $r_S \subseteq r_T$.*

The relationship of *equivalence* between two relations is expressed as two way subsumption relationship: $r_S \equiv r_T$ *iff* $r_S \subseteq r_T$ and $r_T \subseteq r_S$. In this way, we support the computation of both *subsumption* and *equivalence* relationships.

In this work, we consider only entity-entity relations. We do not consider entity-literal relations, because the equivalences between literals are managed differently and are not materialized in KBs. However, once such equivalences are established, one can use our approach to align entity-literal relations.

5 SORAL: Relation Alignment

We propose *SORAL*, an online relation alignment approach, which for a pair of knowledge bases, a source $\mathcal{K}_S$ and a target $\mathcal{K}_T$, works under two assumptions: (i) each KB is accessible through a SPARQL endpoint, (ii) for a KB pair we assume that their entity instances are partially aligned.

The process of discovering *subsumption* and *equivalence* relation alignments in *SORAL* is outlined in two main steps:

(1) **Candidates Generation.** For a relation $r_S \in \mathcal{R}_S$, find all overlapping relations $r_T \in \mathcal{R}_T$ *i.e.* $\exists x, y : r_S(x, y) \in \mathcal{K}_S \wedge r_T(x, y) \in \mathcal{K}_T$.

(2) **Supervised Model.** For every candidate pair $\langle r_S, r_T \rangle$, we classify it as correct or incorrect depending if the subsumption relationship holds or not. We propose a set of features (Sect. 5.3) which we use to learn a supervised machine learning model.

5.1 Candidate Generation

We observe that for a relation r_S its super-relations r_T must have at least one instantiation in common. Thus, they must satisfy the following constraint:

$$\exists x_S, y_S, x_T, y_T : r_S(x_S, y_S) \wedge x_T \equiv x_S \wedge y_T \equiv y_S \wedge r_T(x_T, y_T)$$

Since, the relations reside at different endpoints, for the candidate generation we perform a *federated* query. The conjunction of the first three terms of the expression can be evaluated at $\mathcal{K}_S$. However, for the last term the query is evaluated at $\mathcal{K}_T$. More precisely, we first evaluate the following query at $\mathcal{K}_S$:

```
$Q_1$: SELECT DISTINCT ?x$_T$ ?y$_T$ FROM <$K_S$> WHERE
  { ?$x_S$ $r_S$ ?$y_S$. ?$x_S$ owl:sameAs ?$x_T$. ?$y_S$ owl:sameAs ?$y_T$. }
```

From the query result-set $S_{r_S} = \{(x_1, y_1) \ldots\}$ of Q_1 we discover the *candidate relations* from $\mathcal{K}_T$ for which these pairs are *instantiations*. Note that a tuple ($\mathtt{x}_T$, $\mathtt{y}_T$) can be the instantiation of the inverse of a relation at $\mathcal{K}_T$. Hence, we discover both *direct* and *inverse* relations. For this purpose, we use the *VALUES* feature from SPARQL 1.1 and compute using only one query both inverse and direct relations.

```
$Q_2$: SELECT DISTINCT ?r$_T$ ?d FROM <$K_T$> WHERE {
VALUES (?x ?y) {(x$_1$ y$_1$) ...}
 { SELECT ?x ?r$_T$ ?y ?d WHERE {?x ?r$_T$ ?y. VALUES ?d {"d"}}}
  UNION
 { SELECT ?x ?r$_T$ ?y ?d WHERE {?y ?r$_T$ ?x. VALUES ?d {"i"}}} }
```

The result-set of query $Q2$, $C(r_S)$, denotes the set of candidate relations r_T. For simplicity, we assume that all relations are direct. This is because we have assumed (Sect. 3) that each KB defines for each relation, its inverse.

We avoid the transfer of the entire instantiations of a relation (as it introduces a bottleneck), and instead sample for a minimal set of tuples (x_T, y_T) that are transferred from the source to the target. We discuss the sampling strategies in the following

5.2 Sampling Strategies

In many cases SPARQL endpoints place limitations on the number of triples one can transfer. Even when full access is provided, transferring the full data is expensive (both in terms of time and network bandwidth consumption).

To overcome these issues we suggest three *sampling* strategies for entity instance selection, which we use to generate *relation candidates*. For r_S, the objective is to sample for a minimal set of *representative* instantiations S_{r_S}, such that they provide optimal coverage on discovering super-relations r_T.

First–N Sampling. It is an efficient way to query for the first N returned entity samples (x_T, y_T). The drawback is that it does not provide *representative samples*, due to the fact that it is subject to how the data is added into the

KBs. This represents a baseline to show the impact of sampling strategies on generating accurate relation alignments.

Random Sampling. In this case we use the `RAND()` feature of SPARQL 1.1 to query for representative samples for S_{r_S}. Contrary to first–N, ordering the matching triples through the `RAND()` function is expensive and introduces a significantly higher overhead in the query-execution when performing the relation alignment. Moreover, dependent on the sample size, the samples might be biased towards the classes with more instances, and as such it has an impact in the candidate relations we generate from $\mathcal{K}_T$.

Stratified Sampling. Here, we account for the possibly missing candidate relations from $\mathcal{K}_T$ due to the sample size, for relations whose domain/range belongs to fine grained entity types. Through stratified sampling we can achieve a better coverage in terms of uncovered relation alignments for r_S.

We group entities into homogeneous groups/strata (a group is represented by an entity type) before sampling. Additionally, the groups are further defined at various depths at the type taxonomy levels in order to find out the optimal groups which yield the highest coverage in terms of candidate relations. To ensure the *disjointness* (a prerequisite for stratified sampling), from the *transitive type closure* for an entity, we associate it to its most specialized type (dependent at what level in the type taxonomy we are interested to group entities), thus, ensuring disjointness of groups.

At high levels in the taxonomy, the sampling is similar to random sampling, whereas for deeper levels, we have more groups, and as a consequence more representative samples, leading to an increased coverage of candidate relations from $\mathcal{K}_T$. The query to retrieve the samples is shown below, where N is the sample size for each strata, proportional to the number of entities in it and the given total sample size.

Q_3: `SELECT DISTINCT` $?x_T$ $?y_T$ `WHERE` {
  $?x_S$ r_S $?y_S$. $?x_S$ `owl:sameAs` $?x_T$.
  $?y_S$ `owl:sameAs` $?y_T$. $?x_S$ `rdf:type` $?type$.
} `ORDER BY RAND() LIMIT` N

5.3 Features

In this section, we describe the features we use to learn the supervised model in *SORAL* to predict whether the subsumption relationship for a relation pair holds. We consider features which we categorize into two main categories. Firstly, we consider inductive logic programming (ILP) approaches that work under the *open* and *closed* world assumptions w.r.t the instantiations of relations r_S and r_T. In the second group (GRS), we look into general statistics extracted from relations, specifically we assess the domain/range class distribution of the relations r_S and r_T. It is important to note that the features are computed over the sampled entity instances S_{r_S} and for the candidate relation in $C(r_S)$.

ILP – Features

Closed World Assumption – CWA. Proposed in [10], cwa_{conf} is used to mine association rules and was first used for relation alignment between two KBs in state of the art [28]. It works under the *closed world assumption*, where the KBs are assumed to be complete, hence there are no missing statements. The cwa_{conf} score is computed as in Eq. 1. It measures the overlap in terms of the number of instantiations in common between r_S and r_T, normalised by the number of instantiations of r_S.

$$cwa_{conf}(r_S \Rightarrow r_T) := \frac{overlap(r_S \wedge r_T)}{r_S} \tag{1}$$

We gather such counts by querying the respective SPARQL endpoints of $\mathcal{K}_S$ and $\mathcal{K}_T$.

The cwa_{conf} measure provides strong signal for the alignment of relations in the case when the relations are *complete* (i.e., KBs have all statements for a given relation). However, due to the fact that KBs are constructed from different sources, complementary statements for an entity in different KBs are considered as counter-examples.

Partial Completeness Assumption – PCA. The second feature, pca_{conf}, is an ILP measure proposed in [14] with the purpose of mining *Horn rules* for an input KB. It is able to handle incompleteness in KBs, and works under the *partial completeness assumption.* For a relation r_S and instance x, it is assumed that the KB contains either *all* or *none* of the r_S-triples with x as a *subject.* As counter-examples for the alignment $r_S \Rightarrow r_T$, is considered any pair (x, y) that is an instantiation of r_S but not of r_T. More formally, $counter(r_S \Rightarrow r_T)$ $:= \#(x, y) : r_S(x, y) \wedge \exists y_2, y_2 \neq y : r_T(x, y_2) \wedge \neg r_T(x, y)$.

$$pca_{conf}(r_S \Rightarrow r_T) := \frac{overlap(r_S \wedge r_T)}{overlap(r_S \wedge r_T) + counter(r_S \Rightarrow r_T)} \tag{2}$$

Similar to cwa_{conf}, to compute pca_{conf} we extract the counts for r_S and r_T through SPARQL queries on $\mathcal{K}_S$ and $\mathcal{K}_T$. Note that, pca_{conf} can have maximal score even when the overlap between relation instantiations consists of few tuples with no counter-examples. This is helpful in detecting alignments with a small overlap in the two KBs. On the other hand, it increases the likelihood of getting false positives. These are caused by erroneous facts and incomplete data.

However, considering jointly cwa_{conf} and pca_{conf} features has the effect of *regularizing* each other; for high pca_{conf} scores but low cwa_{conf} the chance of having a correct alignment is low, and vice-versa. We make use of this assumption and compute a *joint probability* score for a relation pair being correct by considering the pca_{conf} and cwa_{conf} scores jointly.

Partial Incompleteness Assumption – PIA. The pca_{conf} may hold for *functional relations* (see Eq. 4), but is less probable for one-to-many relations. The intuition is that if the average number of r_S-triples per subject is high, then it is more likely that not all r_S-triples of some subject x have been extracted.

The same observation holds also for the triples of r_T. Note, that we have yet to prove that r_S is subsumed in r_T. Hence, the counter-examples of less functional relations should be weighted less than the counter-examples of more functional relations. We measure the pia_{conf}score as:

$$pia_{conf}(r_S \Rightarrow r_T) := \frac{overlap(r_S \wedge r_T)}{overlap(r_S \wedge r_T) + (counter(r_S \Rightarrow r_T) * func(r_S))} \tag{3}$$

where, $func(r_S)$ measures the probability that a counter example may not exist.

General Relation Statistics (GRS) – Features. In the GRS feature group, we propose features that take into account the cardinality of the two relations, and the types extracted from the entity instances in the *domains* and *ranges* of a relation.

Functionality of a Relation. A relation is called *functional*, if there are no two distinct facts that share the relation and the first argument. Real life KBs may be noisy, and contain distinct facts for relations that should be functional (such as *bornIn*). Therefore, we make use of the functionality [28]:

$$func(r) := \frac{\#x : \exists y : r(x,y)}{\#(x,y) : r(x,y)} \tag{4}$$

Perfect functional relations have a functionality of 1.

From this intuition we assume that if r_S is subsumed by r_T, then the number of facts per subject in r_S should be lower than in r_T. In other words, the functionality of r_S should be grater or equal to the functionality of r_T, that is if $r_S \Rightarrow r_T \rightarrow func(r_S) \geqslant func(r_T)$. Finally, the functionality feature we consider here is the functionality difference between r_S and r_T, $func_{diff} := func(r_S) - func(r_T) \in (-1, 1)$.

Weighted Jaccard Similarity. We measure the similarity of the type distributions $\mathcal{D}(\cdot)$ between the relations r_S and r_T. We denote with $\mathcal{D}^s(r_S)$ and $\mathcal{D}^s(r_T)$ the type distribution for *subject* entities for r_S and r_T, respectively; each type is represented by the proportion of entities (out of the total entities for a relation).

Since we are interested to find the subsumption $r_S \Rightarrow r_T$, the intuition behind this feature is that the respective type distributions should be similar, or $\mathcal{D}(r_S)$ should be entailed in $\mathcal{D}(r_T)$. In other words, the target relation r_T, should be able to represent entity instances from r_S, namely through its *domain* entity type definition[2].

To compute $\mathcal{D}(\cdot)$, first we unify the type representation of entities in the two KBs by picking one of the taxonomies in either $\mathcal{K}_S$ or $\mathcal{K}_T$. This is necessary, due to the fact that two KBs use different schemas. We are able to do this due to the restriction that we consider only entities that are linked across KBs through `owl:sameAs` statements. The similarity is computed as in Eq. 5.

$$WJ(\mathcal{D}(r_S), \mathcal{D}(r_T)) := \frac{\sum_{t \in \mathcal{D}(r_S) \wedge \mathcal{D}(r_T)} min(\sigma(t^S), \sigma(t^T))}{\sum_{t \in \mathcal{D}(r_S) \wedge \mathcal{D}(r_T)} max(\sigma(t^S), \sigma(t^T))} \tag{5}$$

---

[2] Each relation in a RDFS schema has two properties denoting the *domain* and *range*.

where, $\mathcal{D}(\cdot)$ represents either the type distribution for the subject or object values based on that if the relation is a direct or inverse relation. $\sigma(t)$ represents the score of a type in the distribution for a given relation.

Weighted Jaccard Dissimilarity. Contrary to the previous feature which is computed on overlapping types, in this case, we compute the *weighted dissimilarity score* (WDS) between r_S and r_T. Specifically, if a specific entity type from r_S does not exist in r_T, this accounts for a dissimilarity between the two relations, and lowers the likelihood of r_T subsuming r_S.

$$WDS(\mathcal{D}(r_S), \mathcal{D}(r_T)) := \frac{\sum_{t \in \mathcal{D}(r_S) \wedge t \notin \mathcal{D}(r_T)} \sigma(t^S)}{|\mathcal{D}(r_S) \setminus \mathcal{D}(r_T)|} \tag{6}$$

The intuition is that if r_T cannot represent types from r_S which account to a large proportion of entities, then it is unlikely for $r_S \Rightarrow r_T$ to hold.

ILP Score Relevance Probability. In the case of ILP features, the scores are subject to KB pairs. For specific relations we can achieve nearly perfect pca_{conf}score with low overlap in terms of entity instantiations between relations (e.g. single entity x), yet, with the subsumption relationship unlikely to hold. On the other hand, low pca_{conf}scores may result even if the overlap of instantiations is high in absolute numbers, but due to the complementary nature of datasets might yield to many counter examples.

We circumvent such shortcomings of the ILP features, by assessing the likelihood of the $pca_{conf}$and cwa_{conf}scores for a subsumption relationship to hold between a relation pair. We measure the *prior* probabilities (in our training data) and use these probabilities as features in our learning algorithm. To do so, we first discretize the $pca_{conf}$and cwa_{conf}scores into the ranges with a cut-off point of 0.1, $\{0, 0.1, \ldots, 1.0\}$. For a given pca_{conf}or cwa_{conf}score, we denote with $\langle r_S, r_T \rangle^c$ the set of relation alignment candidates that are correct, and with $\langle r_S, r_T \rangle$ the complete set of relation alignment candidates.

ILP Prior. We measure the *prior probabilities*, $p(correct|pca_{conf})$ or $p(correct|cwa_{conf})$, where the subsumption relationship holds for a pca_{conf}or cwa_{conf}score as the ratio of the cases where the relation holds, divided by the total relation pairs having that discretized score.

Joint PCA and CWA Probability. Here we address the shortcomings of $cwa_{conf}$and pca_{conf}score, by learning a *joint probability* score in which the subsumption relationship holds. We compute the joint relevance probability as following.

$$p(correct|pca_{conf}, cwa_{conf}) :=$$

$$\frac{\#\langle r_S, r_T \rangle^c : pca_{conf}(r_S \Rightarrow r_T) = pca_{conf} \wedge cwa_{conf}(r_S \Rightarrow r_T) = cwa_{conf}}{\#\langle r_S, r_T \rangle : pca_{conf}(r_S \Rightarrow r_T) = pca_{conf} \wedge cwa_{conf}(r_S \Rightarrow r_T) = cwa_{conf}} \tag{7}$$

where for a given discretized $pca_{conf}$and cwa_{conf}score we simply count the number of relation pairs whose scores match and the corresponding alignment is relevant, over the total number of relation pairs with the respective pca_{conf} and cwa_{conf} scores. We expect this feature to be sparse, hence, we use the simple priors as fall-back features.

6 Experimental Setup

Here we present the evaluation setup of *SORAL*. In our setup we host the evaluation datasets in the Virtuoso Universal Server[3] with SPARQL 1.1 support.

Datasets. We evaluate *SORAL* on the following real-world knowledge bases which are the most commonly used datasets from LOD and serve as general purpose datasets.

YAGO (Y). From the YAGO2 [29] dataset we use the *core* and *transitive type* facts, excluding the entity labels given that we consider only entity–entity relations. The size of YAGO is approximately 900 MB.

DBpedia (D). For DBpedia [2] we consider the *entity types* and the *mapping-based properties*, with a size of 5.5 GB.

Freebase (F). For Freebase[4] we take a subset of entities that have `owl:sameAs` links to DBpedia, with a total of 30 GB of data.

Entity-Entity Relations. From the aforementioned KBs, we extract all possible `entity-entity` relations, and filter out those with less than 50 triples. After filtering we are left with 36 relations from YAGO, 563 from DBpedia, and 1666 from Freebase.

Entity Links `owl:sameAs`. We are in hold of `owl:sameAs` links between the pairs DBpedia – YAGO (2.8 MM links), and DBpedia – Freebase (3.8 MM links), in the case of YAGO and Freebase we infer the `owl:sameAs` links (2.7 MM links) through the DBpedia links.

Sampling Strategies Setup. In order to measure the right samples size to have an optimal coverage and maintain the efficiency of our approach, we vary the sample sizes between {100, 500, 1000}. We evaluate all three sampling strategies, and in the case of stratified sampling we construct the strata based on the DBpedia type taxonomy[5]. We take advantage of the fact that all sampled entities from the various KBs have equivalent entities in DBpedia. We opt for the DBpedia taxonomy due to the fact that the types form a hierarchy.

Ground-Truth Construction. Due to the limited resources, the authors of the paper have served as expert annotators and we manually construct the ground-truth for the relation alignment process. The ground-truth is constructed for each KB pair. We guide the process of ground-truth creation by displaying the individual pairs $\langle r_S, r_T \rangle$. For each pair, apart form the labels of the relations,

---

[3] http://virtuoso.openlinksw.com.

[4] http://www.freebase.com.

[5] http://mappings.dbpedia.org/server/ontology/classes/.

the expert annotators need to assess a sample of instantiations for each relation in order to assess correctly whether a relation subsumes another.

Learning Framework. In *SORAL* we feed in all the computed features into a supervised model. In our case we use a *logistic regression* (LR) model [4] (any other model can be used), and learn a binary classifier. For each relation pair $\langle r_S, r_T \rangle$ it outputs either *'correct'* or *'incorrect'* depending whether the subsumption relationship holds in our ground-truth dataset. Finally, we evaluate the performance of our model by considering a 5-fold cross-validation strategy.

Evaluation Metrics. We evaluate *SORAL* on two main aspects: (i) accuracy and (ii) efficiency. In terms of accuracy, we compute standard evaluation metrics, like precision (**P**), recall (**R**) and F-measure (**F1**). For *efficiency* we measure the overhead in the query-execution process in terms of: (i) time (**t**), as the amount of time taken to sample entities for the set S_{r_S}, and (ii) network bandwidth usage (**b**) as the amount of bytes we transfer over the network for the different sample sizes in S_{r_S}. Note here, that a traditional schema-matching approach operating on a dataset snapshot, and as such they introduce an incomparably higher overhead in terms of bytes and time.

Competitors. We compare *SORAL* against two existing approaches. First is a *state of the art* approach in relation alignment, a system called *PARIS* [28], which implements the cwa_{conf}measure. The second competitor, is called *ROSA* [13] and makes use of the pca_{conf}measure. It must be noted that the competitors are unsupervised approaches, and as such do not need training data. However, as it will be shown in our evaluation applying such measures in an unsupervised manner results in poorer performance.

To ensure fairness in our comparison, since the two baselines use specific thresholds for the alignment process, we select their best threshold parameter such that it maximizes the F1 score. The best configuration for *PARIS* is a threshold of cwa_{conf}= 0.1, whereas for *ROSA* a threshold of pca_{conf}= 0.3.

Table 1. The performance of *SORAL* for the different sampling strategies and sizes.

`Sampling`	100			500			1000		
	`P`	`R`	`F1`	`P`	`R`	`F1`	`P`	`R`	`F1`
firstN	.80	.45	.56	.83	.48	.60	.82	.48	.59
random	.80	.45	.56	.81	.44	.56	.82	.44	.56
str.lvl–2	.80	.40	.51	.81	.44	.57	.77	.40	.49
str.lvl–3	.80	.42	.54	**.84**	**.52**	**.64**	.82	.50	.61
str.lvl–4	.79	.44	.55	.82	.49	.61	.80	.49	.59
str.lvl–5	.77	.44	.55	.82	.47	.59	.80	.48	.60

7 Results and Discussion

In this section we present the evaluation results of *SORAL* by assessing two main aspects: (i) the performance for the relation alignment task and its comparison against the competitors, (ii) the *efficiency* by measuring the introduced overhead in terms of time and network bandwidth overload through the sampling strategies.

7.1 Relation Alignment Accuracy

Sampling Configurations. Here, we show the impact of the sampling strategies on the candidate generation process. Through sampling we can *efficiently* perform the alignment process in an online setting, and at the same time provide accurate results.

Table 1 shows the sampling strategies presented in Sect. 5.2. In the case of the *stratified sampling* we construct the strata based on the DBpedia type taxonomy with a depth level from 2 up to 5 (DBpedia has a maximum depth level of 7). The results are averaged across the different KB pairs. Here we aim at finding the best parameter configuration for *SORAL*, specifically for: (i) sampling strategy, and (ii) sample size.

From Table 1 we see that the best performing strategy is based on the *stratified sampling*, with strata constructed at the depth level of three, and with a sample size of 500. With these two parameters, *SORAL* yields an F1 score of 0.64. We note a small fluctuation in terms of precision for the sample sizes of 500 and 1000. We check for statistical significance on the resulting F1 scores, however, the difference proved to be insignificant with a $p - value = .17$. On the other hand, we find statistical significance between the F1 scores for 100 and 500 entity samples ($p - value = .01$).

Table 2. The performance of *SORAL* in comparison to the competing baselines.

	Sampled data									Full data								
	SORAL			ROSA			PARIS			SORAL			ROSA			PARIS		
				$pca_{th} = 0.3$			$cwa_{th} = 0.1$						$pca_{th} = 0.3$			$cwa_{th} = 0.1$		
$\mathcal{K}_S \rightarrow \mathcal{K}_T$	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
YAGO → DBpedia	1.0	.68	**.81**	.17	.75	.28	.71	.66	.68	.92	.73	**.81**	.06	.68	.11	.42	.54	.48
YAGO → Freebase	.87	.67	**.76**	.11	.78	.20	.55	.59	.57	.82	.82	**.82**	.03	1.0	.05	.40	1.0	.57
DBpedia → Freebase	.72	.36	**.48**	.10	.67	.18	.31	.50	.40	.69	.38	**.49**	.05	.85	.09	.31	.65	.42
DBpedia → YAGO	.86	.60	**.71**	.30	.72	.43	.70	.66	.68	.57	.49	**.53**	.18	.55	.27	.40	.45	.43
Freebase → DBpedia	.88	.51	.64	.27	.79	.41	.65	.65	.65	.87	.66	**.75**	.34	.93	.50	.72	.57	.64
Freebase → YAGO	.72	.34	**.46**	.22	.39	.28	.42	.37	.39	.69	.74	**.71**	.61	.86	.71	.73	.60	.66
Average	.84	.52	**.64**	.19	.68	.29	.55	.57	.56	.76	.64	**.69**	.21	.81	.29	.49	.63	.53

SORAL Effectiveness. Table 2 shows the results for the best configuration of *SORAL* (with stratified sampling at depth level 3) for the individual KB pairs. In the "Sampled Data" results group we compare *SORAL* against the competitors *ROSA* and *PARIS*. The results are highly significant when comparing *SORAL* against the competitors for the F1 score. In terms of precision, we have a relative improvement of 336% when compared to *ROSA*, whereas in the case of *PARIS* we have a 51% improvement. In terms of recall, we perform worse, however, the low pca_{conf} and cwa_{conf} thresholds for our baselines ($pca_{th} = 0.3$ and $cwa_{th} = 0.1$), lead to a low precision which makes such results hardly usable. Increasing the thresholds results an increase of precision, however, strongly penalizing the recall of the competitors.

In the group of results in column "Full Data", we show the results of *SORAL* and the competitors when computed without sampling for the relation candidate generation. Here too we outperform the competitors to a large extent, with an average difference of 16% in terms of F1 for *PARIS* and even higher difference of 40% to *ROSA*.

It is interesting to note that for *SORAL* on average we miss 13% of aligned relations from the real coverage performance of *SORAL* when comparing the results with sampling and without. In terms of F1 measure we have only a small difference of 6%, which presents an optimal results when comparing the advantages we get in terms of *efficiency* through sampling.

Finally, we note varying performance of *SORAL* (with and without sampling) across the different KB pairs. For instance, in the case of YAGO → DBpedia we have perfect precision and the best F1 score. It is evident that DBpedia and YAGO represent high quality KBs, whereas Freebase is more noisy, thus, leading to a poorer performance. The main reason for such noise is that relations in Freebase are created by its users, and as such are not unique, in the sense that there are multiple relations representing similar concepts in Freebase.

From Table 2 we further notice that in the case of DBpedia to YAGO we have worse results when comparing the full against results obtained through sampling. This may be caused due to the difference in the number of statements per relation in the two KBs. DBpedia has a larger set of entity instances and statements per relation, and as such the ILP and GRS features are penalized more heavily in the full dataset. We believe that in this case the sampling plays a normalizing factor for such differences in terms of statements and the impact on the feature computation.

Feature Ablation. In the feature ablation we show as to what is attributed the performance of *SORAL*. Figure 2 shows the feature ablation results for the different feature groups in Sect. 5.3. The results are averaged across the different KB pairs and are computed based on models trained on the full data, such that we can assess the true power of the different groups.

The highest impact is attributed to the *GRS* feature group. This follows our intuition where we hypothesized that for a relation to be subsumed in a target relation, one of the important factors is the ability of r_T to represent the entity

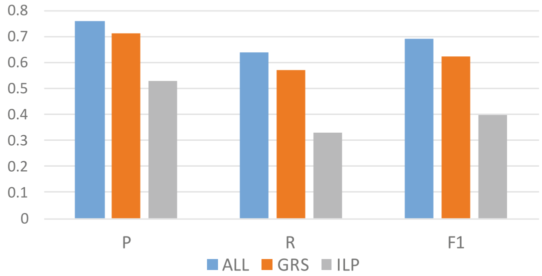

Fig. 2. Feature ablation for the different feature groups.

instances. In other words, the entity types for two relations should be similar. The next assumption we made was that *ILP* feature scores, depending on the KB pairs, sometimes are insufficient, hence, the corresponding likelihood scores of the ILP measures can provide additional information on predicting correctly the label of a relation pair.

7.2 Query-Execution Overhead

Here we show the overhead in terms of *time* and *network bandwidth* usage for the *sampling* process. These represent important aspects considering that our relation alignment is setup in an online setting.

Time overhead. With respect to the *time* efficiency factor, we measured the time it takes to perform the different sampling strategies. The most efficient sampling strategy is *first–N* taking the least amount of time. The *stratified* sampling strategy requires the most amount of time, with a maximum of 3 *s* (for 1000 instances); a significantly higher execution time compared to other strategies. However, such an overhead for the task at hand is arguably not high.

Network bandwidth overhead. Understandably the amount of overhead in terms of bandwidth is uniformly distributed across the different sampling strategies. That is, considering that we sample for the same amount of entity instances. The highest amount of bandwidth overhead is introduced when we sample for 1000 entity instances, with a maximum 140 *kb*. In the performance evaluation of *SORAL* in Table 1, we found the optimal results to be with 500 entity sample instances, resulting in bandwidth overhead of 60 *kb*. This contrary to existing approaches that require the full content of a KB, represents a highly efficient way to compute such alignments.

8 Conclusion

We proposed *SORAL*, a supervised machine learning approach for relation alignment. We perform the specific task where for a given source relation r_S we find relation subsumption in a target KB. Furthermore, we employ *SORAL* in an online setting, where we ensure the *efficiency* of the proposed approach by extracting minimal samples of entity instances for a relation from the respective SPARQL endpoints. To ensure the accuracy of *SORAL* we computed features that range from *association rule mining* features, to *general statistics* from relations.

References

1. Datasets by topical domain. http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC/ISWC - 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). doi:10.1007/978-3-540-76298-0_52
3. Aumueller, D., Do, H.-H., Massmann, S., Rahm, E.: Schema and ontology matching with coma++. In: SIGMOD (2005)
4. Bishop, C.M.: Pattern Recognition and Machine Learning, vol. 1. Springer, Heidelberg (2006)
5. Bizer, C., Heath, T., Idehen, K., Berners-Lee, T.: Linked data on the Web. In: WWW (2008)
6. Böhm, C., de Melo, G., Naumann, F., Weikum, G.: Linda: distributed web-of-data-scale entity matching. In: CIKM (2012)
7. Cheatham, M., Hitzler, P.: String similarity metrics for ontology alignment. In: Alani, H., et al. (eds.) ISWC 2013. LNCS, vol. 8219, pp. 294–309. Springer, Heidelberg (2013). doi:10.1007/978-3-642-41338-4_19
8. Cruz, I.F., Antonelli, F.P., Stroe, C.: Agreementmaker: efficient matching for large real-world schemas and ontologies. PVLDB **2**, 1586–1589 (2009)
9. d'Aquin, M., Adamou, A., Dietze, S.: Assessing the educational linked data landscape. In: WebSci (2013)
10. Dehaspe, L., Toivonen, H.: Discovery of frequent datalog patterns. Data Min. Knowl. Discov. **3**, 7–36 (1999)
11. Dhamankar, R., Lee, Y., Doan, A., Halevy, A.Y., Domingos, P.: imap: discovering complex mappings between database schemas. In: SIGMOD (2004)
12. Doan, A.-H., Madhavan, J., Domingos, P., Halevy, A.: Ontology matching: a machine learning approach. In: Staab, S., Studer, R. (eds.) Handbook of ontologies. International Handbooks on Information Systems, pp. 385–403. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24750-0_19
13. Galárraga, L., Preda, N., Suchanek, F.M.: Mining rules to align knowledge bases. In: AKBC (2013)
14. Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.M.: Amie: association rule mining under incomplete evidence in ontological knowledge bases. In: WWW (2013)
15. Jain, P., Hitzler, P., Sheth, A.P., Verma, K., Yeh, P.Z.: Ontology alignment for linked open data. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010. LNCS, vol. 6496, pp. 402–417. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17746-0_26
16. Kirsten, T., Thor, A., Rahm, E.: Instance-based matching of large life science ontologies. In: Cohen-Boulakia, S., Tannen, V. (eds.) DILS 2007. LNCS, vol. 4544, pp. 172–187. Springer, Heidelberg (2007). doi:10.1007/978-3-540-73255-6_15

17. Koutraki, M., Preda, N., Vodislav, D.: Sofya: Semantic on-the-fly relation alignment. In: EDBT (2016)
18. Koutraki, M., Vodislav, D., Preda, N.: Deriving intensional descriptions for web services. In: CIKM (2015)
19. Koutraki, M., Vodislav, D., Preda, N.: Doris: discovering ontological relations in services. In: ISWC (2015)
20. Lacoste-Julien, S., Palla, K., Davies, A., Kasneci, G., Graepel, T., Ghahramani, Z.: Sigma: simple greedy matching for aligning large knowledge bases. In: KDD (2013)
21. Madhavan, J., Bernstein, P.A., Doan, A., Halevy, A.: Corpus-based schema matching. In: ICDE (2005)
22. Miller, R.J., Haas, L.M., Hernández, M.A.: Schema mapping as query discovery. In: VLDB (2000)
23. Movshovitz-Attias, D., Whang, S.E., Noy, N., Halevy, A.: Discovering subsumption relationships for web-based ontologies. In: Proceedings of the 18th International Workshop on Web and Databases (2015)
24. Parundekar, R., Knoblock, C.A., Ambite, J.L.: Linking and building ontologies of linked data. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) ISWC 2010. LNCS, vol. 6496, pp. 598–614. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17746-0_38
25. Schmachtenberg, M., Bizer, C., Paulheim, H.: Adoption of the linked data best practices in different topical domains. In: Mika, P., et al. (eds.) ISWC 2014. LNCS, vol. 8796, pp. 245–260. Springer, Cham (2014). doi:10.1007/978-3-319-11964-9_16
26. Seligman, L., Mork, P., Halevy, A.Y., Smith, K.P., Carey, M.J., Chen, K., Wolf, C., Madhavan, J., Kannan, A., Burdick, D.: Openii: an open source information integration toolkit. In: SIGMOD (2010)
27. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Trans. Knowl. Data Eng. **25**, 158–176 (2013)
28. Suchanek, F.M., Abiteboul, S., Senellart, P.: Paris: probabilistic alignment of relations, instances, and schema. PVLDB **5**(3), 157–168 (2011)
29. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a core of semantic knowledge - unifying WordNet and Wikipedia. In: WWW (2007)
30. Udrea, O., Getoor, L., Miller, R.J.: Leveraging data and structure in ontology integration. In: SIGMOD (2007)
31. Wang, S., Englebienne, G., Schlobach, S.: Learning concept mappings from instance similarity. In: Sheth, A., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 339–355. Springer, Heidelberg (2008). doi:10.1007/978-3-540-88564-1_22
32. Wijaya, D.T., Talukdar, P.P., Mitchell, T.M.: Pidgin: ontology alignment using web text as interlingua. In: CIKM (2013)