Statistical Induction of Coupled Domain/Range Restrictions from RDF Knowledge Bases

Basil Ell, Sherzod Hakimov and Philipp Cimiano

CIT-EC, Universität Bielefeld
{bell, shakimov, cimiano} @cit-ec.uni-bielefeld.de

Abstract. Statistical Schema Induction can be applied on an RDF dataset to induce domain and range restrictions. We extend an existing approach that derives independent domain and range restrictions to derive coupled domain/range restrictions, which may be beneficial in the context of Natural Language Processing tasks such as Semantic Parsing and Entity Classification. We provide results from an experiment on the DBpedia graph. An evaluation shows that high precision can be achieved.

Keywords: Statistical Schema Induction, RDF, property restrictions

1 Introduction

RDFS domain restrictions and range restrictions for a property let us infer to which class the subject and the object, respectively, of a triple with this property belong to. For example, given the domain restriction `(foaf:knows, rdfs:domain, foaf:Person)` and `(foaf:knows, rdfs:range, foaf:Person)`, from the triple `(ex:Frank, foaf:knows, ex:Vidya)` we can infer the class assertions `(ex:Frank, rdf:type, foaf:Person)` and `(ex:Vidya, rdf:type, foaf:Person)`.

In the context of the Semantic Web knowledge representation format RDF, *Statistical Schema Induction* is the process of inducing ontological statements such as RDFS statements and OWL statements from RDF data, such as domain and range restrictions or subclass relations.

Domain restrictions and range restrictions were so far considered independently [9]. For example, RDFS does not allow to specify that given a statement `(ex:s, ex:p, ex:o)`, if `ex:s` belongs to class c_1, then `ex:o` belongs to class c_2. This makes sense since property restrictions are entailment rules and not constraints. However, when applying them as heuristics instead, coupling domain and range restrictions becomes interesting. For example, consider the DBpedia property `dbo:champion` with domain `dbo:SportsEvent` and range `dbo:Athlete`. When observing concrete data, one can see that this property is either used for subjects of class `dbo:SportsEvent` and objects of class `dbo:Athlete` or, among other cases, for subjects of class `dbo:GolfTournament` and objects of class `dbo:GolfPlayer`. Having identified a golf tournament in text near and entity identified as a person, one may now guess that the person is a golf player.

In this paper we propose an approach based on Frequent Itemset Mining (FIM) to statistically induce coupled domain and range restrictions. Given a knowledge base encoded in RDF, for a property p our method creates a set of statements of the form (D, R, c) where D is the set of domain classes, R is the set of range classes, and $c \in [0, 1]$ is a support value.

We envision that the application of coupled domain/range restrictions is interesting in the following two scenarios.

1. Semantic Parsing: Question answering may consist of the task of mapping natural language questions to SPARQL queries that can then be evaluated over an RDF dataset. An example of a question and the corresponding query from QALD-6[1] is given below: "Which actors were born in Germany?"

   ```
   PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
   PREFIX dbo: <http://dbpedia.org/ontology/>
   PREFIX dbr: <http://dbpedia.org/resource/>
   SELECT ?uri WHERE {
       ?uri rdf:type dbo:Actor .
       ?uri dbo:birthPlace dbr:Germany .
   }
   ```

   Listing 1. SPARQL query from QALD-6

   The string "born" can be interpreted as mapping to a property. In DBpedia there are various candidates for the interpretation of "born": `dbo:birthDate`, `dbo:birthYear`, and `dbo:birthPlace`. For the question above, only the interpretation in terms of `dbo:birthPlace` makes sense as `Germany` matches to the range restriction of `dbo:birthPlace` (since `dbr:Germany` is a `dbo:Place`). As the entity `dbr:Germany` does not comply with the range restrictions of `dbo:birthYear` nor of `dbo:birthDate`, these interpretations can be ruled out by a Question Answering system that takes into account domain and range restrictions. Using coupled domain/range restrictions can help to rule out further interpretations in terms of properties and entities.
2. Entity Classification: given are a property with two different pairs of domain/range classes: (c_1, c_2) and (c_3, c_4). If we find an entity, e.g. in text, that belongs to c_1, then, if the object of the relation is also found in text but is ambiguous since there are multiple entities that match, then the entity that belongs to class c_2 is more likely to be the correct one than entities of other types, e.g., c_4.

The main contributions of this paper are:

- We motivate the idea of coupled domain/range restrictions that can serve as heuristics in NLP tasks such as Question Answering and Entity Classification.
- We formalize an existing approach for the induction of independent domain and range restrictions as well as our approach for the induction of coupled domain/range restrictions.
- We carry out an experiment on DBpedia and make all induced (independent and coupled) restrictions available to the community.

[1] See http://qald.sebastianwalter.org/index.php?x=benchmark&q=6

The remainder of this paper is structured as follows. We give a quick introduction to Frequent Itemset Mining as well as relevant vocabulary (RDF, RDFS, and OWL) in Section 2, present the existing method to derive domain and range restrictions independently as introduced by [9] as well as our method in Section 3, describe an experiment to induce coupled domain range restrictions from DBpedia in Section 4, evaluate the outcome of the experiment in Section 5, discuss related work in Section 6, and, finally, conclude in Section 7.

All code and data is available at a dedicated page:
https://github.com/ag-sc/SchemaInduction

2 Preliminaries

In this section we briefly introduce the graph-based data model RDF, relevant terms from the RDF(S) vocabularies[2] and from the OWL[3] vocabulary, as well as Frequent Itemset Mining.

2.1 RDF, RDFS, and OWL

An RDF graph consists of a set of *RDF triples*. An RDF triple $t = (s, p, o) \in (\mathcal{U} \cup \mathcal{B}) \times (\mathcal{U}) \times (\mathcal{U} \cup \mathcal{B} \cup \mathcal{L})$ is an ordered set consisting of a subject s, a predicate p, and an object o. $\mathcal{U}$ is a set of URIs, B is a set of blank nodes (existentially qualified variables), and L is a set of literals. $\mathcal{U}$, $\mathcal{B}$, and $\mathcal{L}$ are pairwise disjunct.

From the RDF, RDFS, and OWL vocabularies a small set of terms is relevant in the context of the current work: `rdf:type`, `rdfs:domain`, `rdfs:range`, `rdfs:subClassOf`, and `owl:equivalentClass`.

The property `rdf:type` is used for explicit class assertions. For example, the RDF triple `(ex:X, rdf:type, ex:C)` explicitly expresses that `ex:X` is a member of the class `ex:C`. With the triples `(ex:P, rdfs:domain, ex:C)` and `(ex:P, rdfs:range, ex:D)` a domain restriction (first triple) and a range restriction (second triple) are specified. Given these domain and range restrictions, from a statement, such as `(ex:A, ex:P, ex:B)`, the two class assertions `(ex:A, rdf:type, ex:C)` and `(ex:B, rdf:type, ex:D)` can be derived (by taking into account RDFS semantics[4]) – the first triple via the domain restriction and the second triple via the range restriction. Subclass relations between classes can be expressed using the property `rdfs:subClassOf`. For example, from a statement `(ex:C, rdfs:subClassOf, ex:D)` it follows that `ex:C` is a subclass of `ex:D`, which means that every entity that is a member of class `ex:C` is also a member of class `ex:D`. Equivalence between two classes can be expressed via the symmetric property `owl:equivalentClass`, as in `(ex:C, owl:equivalentClass, ex:D)`.

[2] See http://www.w3.org/TR/rdf-primer/
[3] See http://www.w3.org/TR/owl-features/
[4] See http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

2.2 Frequent Itemset Mining

Let I be a set of items. Given a list T of subsets of the item set (also known as a set of transactions) and given a value $\tau \in [0, 1]$ (also known as the support threshold), the objective of frequent itemset mining consists of creating a set of itemsets O where each set $o \in O$ has a support of greater than or equal to τ, which means that it is subset to at least $\tau * |T|$ itemsets in T. Support of an itemset s is calculated as the number of itemsets $t \in T$ where $s \subseteq t$ devided by $|T|$. See [7] for a textbook-style introduction to Frequent Itemset Mining.

As a classical example, let I be the set of articles available in a grocery store and let T be the set of sets of articles in individual shopping baskets. Frequent Itemset Mining [2] applied on T then tells us which articles are frequently (depending on the τ value) purchased together. Given $\tau = 0.9$, each set of articles can be found in at least 90% of the shopping baskets.

Note that for each frequent itemset, all of its subsets are also frequent itemsets. This property is referred to as the monotonicity of frequent itemsets. *Frequent Maximal Itemset Mining* is the task of deriving only those frequent itemsets that are maximal, which means that a frequent maximal itemset is not a true subset of another frequent itemset.

3 Method

In this section we describe two methods. The first method is the state-of-the-art method by Völker and Niepert [9] that derives independent domain and range statements from an RDF graph using the Frequent Itemset Mining tool [6]. The second method, which is a new contribution, derives coupled domain/range statements from an RDF graph using Frequent Itemset Mining.

Note that we do not intend to provide a method that outperforms the one presented in [9] but rather introduce the new problem of inducing coupled domain/range restrictions for which we introduce a method that builds on the method by Völker and Niepert.

Input to both methods is an RDF graph G, a set of properties P, and a support threshold value τ.

Output for the first method is a set of `(domain classes, support)` tuples and a set of set of `(range classes, support)` tuples. Output for the second method is a set of `(domain classes, range classes, support)` tuples.

The examples that we present to illustrate the method were created by querying against the DBpedia [1] dataset.[5] We ignore classes that are not in the DBpedia namespace, such as `owl:Thing`.

3.1 Deriving Independent Domain and Range Statements

Here we describe the core of the method based on Frequent Itemset Mining to induce domain and range axioms as introduced by Völker and Niepert [9]. The

[5] The prefixes `dbo` and `dbr` refer to `http://dbpedia.org/ontology/` and `http://dbpedia.org/resource/`, respectively.

method to gather data from an endpoint is a bit different, as we will explain later in Section 6.

We create a class dictionary D which is an injection that assigns an integer value to each class in G. Given an RDF graph G, the class dictionary D, and a set of properties P, for each property $p \in P$ we create two transaction tables T_d^p and T_r^p as follows. For each triple $(s, p, o) \in G$ we add a row to T_d^p containing all members of c_s, which is the set of classes the entity s belongs to. We do not add a row to T_d^p if c_s is empty. Furthermore, we add a row to T_r^p containing all members of c_o, which is the set of class identifiers of the classes the object entity o belongs to or the set of datatypes the object literal o belongs to. We do not add a row to T_r^p if c_o is empty. The sets of classes do not only contain the explicit classes c of an entity e as defined via the triple `(e, rdf:type, c)`, but also all superclasses of c via the transitive relation *rdfs:subClassOf*, all their equivalent classes via the transitive relation *owl:equivalentClass*, as well as all implicit types that can be inferred via existing domain or, respectively, range restrictions.

For example, given the property `dbo:author` and the triple `(dbo:Gantenbein, dbo:author, dbr:Max_Frisch)`, for the resource `dbo:Gantenbein` we obtain the set c_s = `{dbo:Book, dbo:Work, dbo:WrittenWork}`. All three classes happen to be available via direct class assertions. `dbo:Work` is the domain of the property (given in the DBpedia Ontology). The superclasses of the directly asserted classes are `dbo:Book`, `dbo:Work`, and `dbo:WrittenWork` and the superclass of the domain class is `dbo:Work`.

For the resource `dbr:Max_Frisch` we obtain the set c_o = `{dbr:Writer, dbo:Person, dbo:Agent}`. Again, all three classes happen to be available via direct class assertions. `dbo:Person` is the range of the property. The superclasses of the directly asserted classes are `dbo:Writer`, `dbo:Agent`, and `dbo:Person` and the superclasses of the range class are `dbo:Person` and `dbo:Agent`. To the domain transaction table $T_d^{dbo:author}$ we would therefore add a line such as `"0 1 2"`, given that the class identifiers refer to the classes `dbo:Book`, `dbo:Work`, and `dbo:WrittenWork`, respectively, in the class dictionary D. This line in $T_d^{dbo:author}$ would express that there is a triple with the property `dbo:author` where the subject belongs to the classes with the identifiers 0, 1, and 2.

Given a transaction table and a support threshold τ, we perform frequent maximal[6] itemset mining to derive a set of classes and their support values where the support values are not less than τ. We reduce each set of classes so that for each class all of its superclasses are removed from the set. Moreover, if a set contains multiple equivalent classes, then all but the first in lexicographic order are removed. For example, the set `{dbo:Athlete, dbo:Person, dbo:Agent}` is reduced to `{dbo:Athlete}`, since this class is a subclass of the two other classes. The purpose of adding superclasses and equivalent classes in the first place is, that within a

[6] Note that the authors of [9] do not explicitly mention that they derive frequent *maximal* itemsets only. But since non-maximal itemsets, such as empty itemsets, are irrelevant, we assume they perform frequent maximal itemset mining.

knowledge graph such as DBpedia, not all entities are necessarily consistently typed. For example, sometimes a superclass is explicitly given, sometimes it is not. Adding them leads to more consistent entries in the transaction table.

For example, given a domain transaction table $T_d^{dbo:author}$ that was created with 10,000 triples and $\tau = 0.5$ we obtained three frequent maximal itemsets:

- `({dbo:Work}, 10000/10000)`
- `({dbo:WrittenWork}, 7771/10000)`
- `({dbo:Book}, 6396/10000)`

From these we can create three domain restrictions:

- `(dbo:author, rdfs:domain, dbo:Work)`
- `(dbo:author, rdfs:domain, dbo:WrittenWork)`
- `(dbo:author, rdfs:domain, dbo:Book)`

Note that this set of domain restrictions contains redundancies. Given the third restriction, the first two restrictions could automatically be created since `dbo:Work` and `dbo:WrittenWork` are superclasses of `dbo:Book`. Therefore, we reduce the output to the restriction with the most specific class. Thus, the restriction `(dbo:author, rdfs:domain, dbo:Book)` is the only itemset remaining after reduction.

3.2 Deriving Coupled Domain and Range Statements

We create a class dictionary D which is an injection as follows. For each class c in G we create two new strings `"domain=" + c` and `"range" + c` via string concatenation and assign different integer values in the dictionary. Given an RDF graph G, the class dictionary D, and a set of properties P, for each property $p \in P$ we create one transaction table T_{dr}^p as follows. For each triple $(s, p, o) \in G$ we add a row (transaction) to T_{dr}^p containing all members of $c_s' \cup c_o'$ As for the other method above, c_s is the set of classes the entity s belongs to and c_o is the set of classes the object entity o belongs to or the set of datatypes the object literal o belongs to and the sets c_s' and c_o' are derived from c_s and c_o, respectively, as follows. For each member $c \in c_s$ (c_o, respectively), we concatenate the string `"domain="` (`"range="`, respectively) and c and add the resulting string to c_s' (c_o', respectively). The sets of classes do not only contain the explicit classes c of an entity e as defined via the triple `(e, rdf:type, c)`, but also all superclasses of c via the transitive relation *rdfs:subClassOf*, all their equivalent classes via the transitive relation *owl:equivalentClass*, as well as all implicit types that can be inferred via existing domain or, respectively, range restrictions.

From a transaction table T_{dr}^p we derive frequent maximal itemsets and reduce them as carried out in the method above. Depending on whether the class names begin with the string `"domain="` or `"range="` we can distribute them to the set of domain classes and the set of range classes.

An example of a frequent itemset is `(dbo:bronzeMedalist, {dbo:Olympic Event}, {dbo:Person}, 5200/10000)`. From this itemset we can create two restrictions: `(dbo:bronzeMedalist, rdfs:domain, dbo:OlympicEvent)` and `(dbo: bronzeMedalist, rdfs:range, dbo:Person)`. However, if these restrictions are represented in this form and are added to an RDF graph, then the domain restrictions and the range restrictions are not coupled anymore.

4 Experiment

For our experiment we set up a SPARQL endpoint containing DBpedia (version 2015-10). The repository contains 8.8 billion triples, 739 classes, 1099 object properties, and 1734 datatype properties.

For some properties DBpedia already contains domain and range restrictions, as listed in Table 1. The headers of the table denote whether properties have domain or range restrictions, e.g., as in the example $D \wedge \neg R$, that the property has a domain restriction but no range restriction.

Table 1. Statistics about Object & Data Properties in DBpedia version 2015-10.

Property Type	Total	$D \wedge R$	$\neg D \wedge R$	$D \wedge \neg R$	$\neg D \wedge \neg R$
Object Properties	1099	704	120	206	69
Datatype Properties	1734	1497	237	0	0

We induced restrictions for 1099 object properties and for 1734 datatype properties (see Table 1) for values of τ in the range `{1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1}` independently of whether domain or range restrictions already exist in DBpedia. Figure 1 shows: i) the number of domain and range itemsets that were induced for different values of τ (e.g., 2290 domain itemsets (= domain restrictions) were induced for $\tau = 1.0$ and 920 range itemsets (= range restrictions) were induced for $\tau = 0.9$), ii) the minimum, average, and maximal number of classes in itemsets in domain and range, and iii) the minimum, average, and maximum number of itemsets induced per property. It is no surprise that the number of domain itemsets and range itemsets grows when τ is decreased. Also, the number of classes within a frequent itemset and the number of frequent itemsets per property grow when τ is decreased.

Results from inducing coupled domain/range restrictions are shown in Figure 2. In detail, it shows: i) the number of itemsets that were induced for different values of τ (e.g.,), ii) the minimum, average, and maximal number of classes in itemsets in domain and range (e.g., 3203 itemsets (= coupled domain/range restrictions were induced for $\tau = 0.9$), *minD*, stands for the minimum number of classes in the sets of domain classes), and iii) the minimum, average, and maximum number of itemsets induced per property.

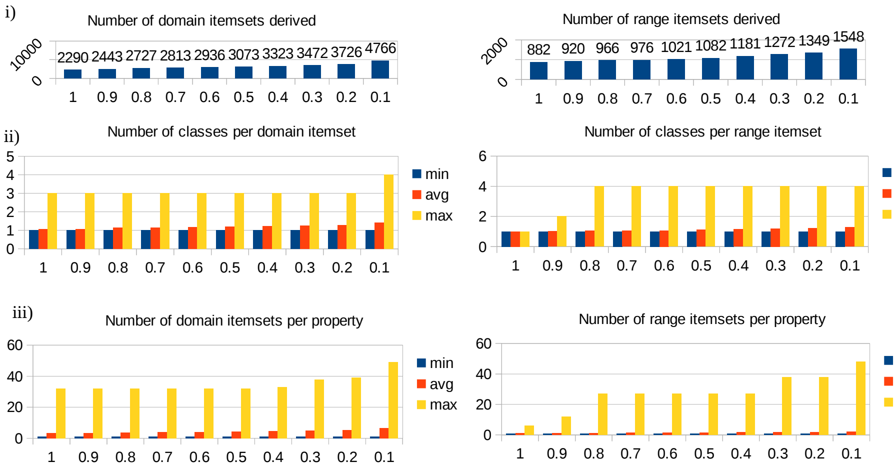

Fig. 1. Basic statistics of induced independent domain and range restrictions.

5 Evaluation

We evaluate the results of both methods:

1. We compare the induced domain (range) restrictions against the domain (range) restrictions that already exist in DBpedia, which we treat as gold standard.
2. We compare the induced coupled domain/range restrictions against the independent domain and range restrictions that already exist in DBpedia, which we treat as gold standard.

To compare against the gold standard of DBpedia, we selected 704 object properties and 1497 datatype properties for which both domain restriction and range restriction are known.

5.1 Evaluation of independent domain and range restrictions

For each domain (range) restriction for which a gold domain (range) restriction exists we compare the induced set of classes against the gold set of classes. This comparison may result in four cases: i) the sets of classes are identical, ii) all induced classes are more specific than all gold classes, iii) all induced classes are less specific than all gold classes, and iv) else (some classes are more specific, some are less specific, some are neither less nor more specific when compared to the gold set). In Table 2 these cases are denoted with $=$, $>$, $<$, and x, respectively.

We define $precision@\tau$ as the number of frequent maximal itemsets derived with the support threshold τ where either the set of induced classes is the same as the gold classes or where the induced classes are less specific, divided by the number of all frequent maximal itemsets derived with the support threshold τ. The arrows ($\uparrow$) in Table 2 mark the columns where this is the case (i.e, $=$ and $<$).

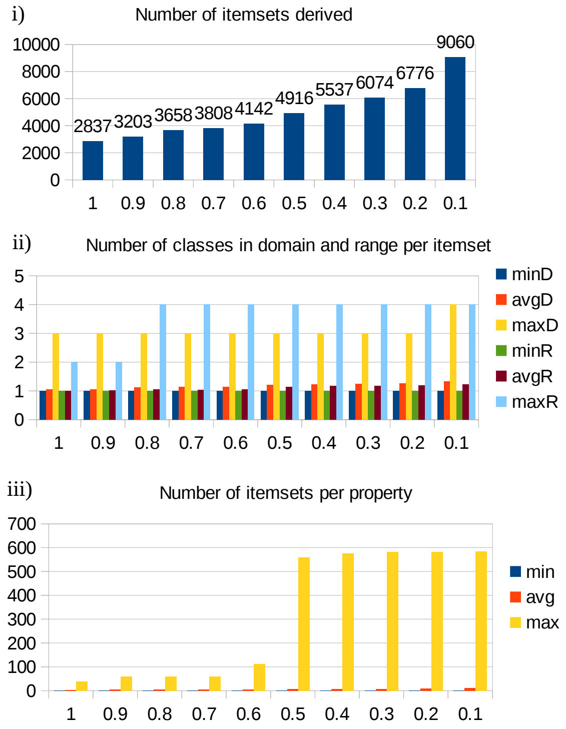

Fig. 2. Basic statistics of induced coupled domain/range restrictions.

Table 3 shows an example for each of the four cases i) the induced domains and the gold domains are identical, ii) the induced domains are more specific than gold domains, iii) the induced domains are less specific than the gold domains, and iv) otherwise, denoted with =, >, <, and x, respectively.

5.2 Evaluation of coupled domain/range restrictions

Induced coupled domain/range restrictions were evaluated similarly to the independent domain and range restrictions. However, the number of cases that may occur is not 4 ($\{=,>,<,x\}$) but instead 4*4 ($\{=,>,<,x\} \times \{=,>,<,x\}$). Table 4 shows an example for each of the 16 possible cases of itemsets with coupled domain and range.

We define *precision*@τ as the number of frequent maximal itemsets derived with the support threshold τ where either the set of induced domain classes is the same or less specific than the gold domain classes and where the induced range classes is the same or less specific than the gold range classes, divided by the number of all frequent maximal itemsets derived with the support threshold τ. The arrows (↑) in Table 5 mark the columns where this is the case (i.e., ==, =<, <=, and <<). Table 6 shows the precision@τ values.

Table 2. Induced independent domain and range restrictions, the frequency of cases, and precision values.

	domain				range				domain precision@τ	range precision@τ
τ	$=$	$>$	$<$	x	$=$	$>$	$<$	x		
1	722	152	1254	162	723	4	155	0	0.86	1.00
0.9	723	249	1254	217	723	20	155	22	0.81	0.95
0.8	723	268	1254	482	723	30	155	58	0.72	0.91
0.7	723	294	1252	544	723	40	155	58	0.70	0.90
0.6	723	319	1252	642	723	52	155	91	0.67	0.86
0.5	723	349	1252	748	723	57	155	147	0.64	0.81
0.4	723	393	1253	954	723	62	155	241	0.59	0.74
0.3	723	424	1254	1071	723	78	155	316	0.57	0.69
0.2	723	477	1254	1272	723	94	155	377	0.53	0.65
0.1	723	574	1248	2221	722	144	151	531	0.41	0.56
	↑		↑		↑		↑			

Table 3. Examples of induced domain classes and the corresponding gold domain classes. The entries for the column *case* correspond to the comparison of induced domains to gold domains, where four cases are possible: = (the induced domain classes are identical with the gold domain classes), > (the induced domain classes are more specific than gold domain classes), < (the induced domain classes are less specific than gold domain classes), and x (otherwise).

case	τ	property	support	induced domain	gold domain
$=$	1.0	`dbo:leftTributary`	4881/4881	`dbo:River`	`dbo:River`
$>$	0.5	`dbo:composer`	5847/10.000	`dbo:TelevisionShow`	`dbo:Work`
$<$	1.0	`dbo:chef`	54/54	`dbo:ArchitecturalStructure`	`dbo:Restaurant`
x	0.8	`dbo:launchSite`	45/509	`dbo:MeanOfTransportation`	`dbo:SpaceMission`

Note that the precision of induced coupled restrictions (Table 2) tends to be below the precision of induced independent restrictions (Table 6). The main reason for the lower precision is that domain and range classes of coupled domain/range restrictions are often more specific than the gold domain and range classes. See for example Table 4 case >>. Adding restrictions with classes that are too specific to a knowledge graph would result in wrong entailment. For example, by adding the restriction `(dbo:champion, rdfs:range, dbo:GolfPlayer)`, from every triple with that property we can then infer via RDFS entailment that the object of the triple is an instance of the class `dbo:GolfPlayer` – in other words: every champion is a golf player. However, these specific classes are helpful as heuristics for certain Natural Language Processing tasks as motivated in the introduction.

Table 4. Examples of induced coupled domain and range classes and the corresponding gold classes. The namespace prefix (dbo) has been consistently omitted. d corresponds to domain restriction and r to range restriction. The entries for the column *case* correspond to the comparison of induced domains to gold domains and induced ranges to gold ranges, respectively, where for each comparison four cases are possible: = (the induced classes are identical with the gold classes), > (the induced classes are more specific than gold classes), < (the induced classes are less specific than gold classes), and x (otherwise).

case	τ	property	support	induced coupled dom./range	gold domain/range
xx	0.3	launchSite	156/509	d=SocietalEvent, MeanOfTransportation r=MilitaryStructure	d=SpaceMission r=Building
$x<$	0.8	launchSite	455/509	d=SpaceMission, MeanOfTransportation r=ArchitecturalStructure	d=SpaceMission r=Building
$x>$	0.1	writer	1195/10000	d=Wikidata:Q11424,Work r=Writer	d=Work r=Person
$x=$	0.4	militaryBranch	4065/10000	d=Organisation, MilitaryPerson r=MilitaryUnit	d=MilitaryPerson r=MilitaryUnit
$<x$	0.3	militaryBranch	5042/10000	d=Agent r=MilitaryUnit,Place	d=MilitaryPerson r=MilitaryUnit
<<	0.7	champion	1349/1349	d=SocietalEvent r=Agent	d=SportsEvent r=Athlete
<>	0.2	militaryBranch	10000/10000	d=Person r=Agent	d=MilitaryPerson r=MilitaryUnit
<=	0.1	militaryBranch	10000/10000	d=Agent r=MilitaryUnit	d=MilitaryPerson r=MilitaryUnit
$>x$	0.3	dam	198/450	d=RaceHorse r=RaceHorse,Eukaryote	d=Animal r=Animal
><	0.2	dam	450/450	d=Mammal r=Species	d=Animal r=Animal
>>	0.9	champion	1252/1349	d=Tournament r=GolfPlayer	d=SportsEvent r=Athlete
>=	0.8	champion	1348/1349	d=GolfTournament r=Athlete	d=SportsEvent r=Athlete
$=x$	0.3	launchSite	191/509	d=SpaceMission r=MilitaryStructure	d=SpaceMission r=Building
=<	0.2	dam	439/450	d=Mammal r=Mammal	d=Animal r=Animal
=>	0.1	dam	439/450	d=Animal r=Horse	d=Animal r=Animal
==	1.0	launchSite	509/509	d=SpaceMission r=Building	d=SpaceMission r=Building

Table 5. Induced coupled domain/range restrictions and the frequency of cases.

τ	==	=>	=<	= x	>=	>>	><	> x	<=	<>	<<	< x	x =	x >	x <	xx
1.0	721	4	157	0	153	4	32	0	1252	10	326	0	165	0	13	0
0.9	723	20	156	22	249	20	56	16	1256	51	327	66	219	0	18	4
0.8	723	30	157	58	269	30	59	24	1254	75	326	134	481	0	34	4
0.7	723	40	155	58	294	35	67	24	1256	101	326	135	545	1	44	4
0.6	723	52	156	91	319	39	69	25	1257	127	327	207	643	0	99	8
0.5	723	58	156	147	350	42	75	48	1256	140	330	306	749	22	122	392
0.4	723	62	158	242	393	50	78	88	1254	144	326	438	954	23	164	440
0.3	723	78	158	316	425	68	92	91	1256	159	328	520	1071	74	186	529
0.2	723	94	156	378	479	78	105	101	1255	203	326	672	1272	135	196	603
0.1	722	146	152	531	574	105	120	156	1250	314	311	1084	2225	239	348	783
	↑		↑						↑		↑					

Table 6. Precision@τ for induced coupled domain/range restrictions.

τ	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
precision@τ	0.87	0.77	0.67	0.65	0.59	0.50	0.44	0.41	0.36	0.27

6 Related Work

Our approach is an extension of a method for statistical schema induction from RDF data introduced by Völker and Niepert [9] which we discuss in detail in Section 3.1. Besides inducing domain and range restrictions of properties, in their work further axioms are induced such as subsumption axioms (e.g., (`ex:A, rdfs:subClassOf, ex:B`)) and transitivity axioms (e.g., (`ex:P, rdf:type, owl:TransitiveProperty`)). This work was subsequently extended in [4] and [5] with further types of axioms. The main difference to this work is that we induce independent domain and range restrictions as well as coupled domain/range restrictions whereas in these works only independent domain and range restrictions are induced. However, all methods are based on Frequent Itemset Mining and only differ in some technical details, such as, how the set of classes an entity belongs to are created (e.g., whether equivalent classes (via `owl:equivalentClass`), superclasses (via `rdfs:subClassOf`), and existing domain and range restrictions are regarded). In both approaches the sets of classes are approximated and it cannot be guaranteed that the sets are complete since data is accessed via SPARQL only. As long as endpoints do not support RDFS and OWL entailment, or as long as not all inferences are materialized, some classes may be missing.

Instead of inducing domain and range restrictions from RDF data, restrictions can also be induced from unstructured data. In [3], Cimiano et al. derive the classes of arguments of verbs from natural language text, which can be seen as a subtask in the context of ontology learning from text. Since in their work verbs are interpreted as binary relations, what they induce are domain and range restrictions. Given an existing taxonomy, for the domain and the range of a property the appropriate level in the taxonomy is selected considering the classes' conditional probabilities. Interestingly, the authors note that "*the domain and range of a relation can actually not be regarded as independent from each other,*" which is exactly what we do in this paper. However, due to a lack of training data, they refrain from regarding coupled domain/range restrictions.

In [8], Töpper et al. induce domain and range restrictions as well as class disjointness axioms from RDF data for the purpose of enabling to detect logical inconsistencies via reasoning. In their work, the domain (range) of a property is the class that most of the subjects (objects) in triples with this property belong to. This has the drawback that if for a property entities belonging to several diverse classes appear in subject (object) position, then the induced domain (range) restriction only regards the most specific superclass of these classes. While this is not wrong, for Natural Language Processing more fine-grained domain and range restrictions are interesting.

7 Conclusion

In this paper we presented the concept of coupled domain/range restrictions and presented an approach to apply Frequent Itemset Mining to induce independent domain and range restrictions as well as coupled domain/range restrictions from an RDF graph. Experiments carried out with the DBpedia dataset showed that high precision can be achieved. We believe that these restriction statements can be beneficially applied in Natural Language Processing scenarios such as Semantic Parsing, Question Answering, and Entity Classification. Therefore, all data obtained as well as our implementation is available to the community on a dedicated website.

Acknowledgements

This work was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

References

1. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In *6th International Semantic Web Conference (ISWC 2007)*, pages 722–735. Springer, 2007.
2. C. Borgelt. Frequent item set mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(6):437–456, 2012.
3. P. Cimiano, M. Hartung, and E. Ratsch. Finding the appropriate generalization level for binary ontological relations extracted from the genia corpus. In *5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 161–169. Citeseer, 2006.

4. D. Fleischhacker and J. Völker. Inductive Learning of Disjointness Axioms. In *On the Move to Meaningful Internet Systems (OTM 2011)*, pages 680–697. Springer, 2011.
5. D. Fleischhacker, J. Völker, and H. Stuckenschmidt. Mining RDF Data for Property Axioms. In *On the Move to Meaningful Internet Systems (OTM 2012)*, pages 718–735. Springer, 2012.
6. P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C.-W. Wu, V. S. Tseng, et al. SPMF: a Java open-source pattern mining library. *Journal of Machine Learning Research*, 15(1):3389–3393, 2014.
7. J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining of Massive Datasets, 2nd Ed.* Cambridge University Press, 2014.
8. G. Töpper, M. Knuth, and H. Sack. DBpedia Ontology Enrichment for Inconsistency Detection. In *8th International Conference on Semantic Systems 2012 (i-Semantics 2012)*, pages 33–40. ACM, 2012.
9. J. Völker and M. Niepert. Statistical Schema Induction. In *8th Extended Semantic Web Conference (ESWC 2011)*, pages 124–138. Springer, 2011.