Semantic Relation Composition in Large Scale Knowledge Bases

Kristian Kolthoff, Arnab Dutta

Data and Web Science Research Group, University of Mannheim, Germany

Abstract. Semantic relation composition is a generalized approach for finding conjunctive relation paths in a knowledge base (KB). In semantic web, direct and inverse relationships between entities provide us with ample of explicit knowledge. But there is a plethora of implicit knowledge beyond these direct paths. Consider a knowledge graph, we can achieve deeper insights about a particular entity if we consider the information shared by its neighboring entities via its adjacent relation paths of arbitrary lengths. In this paper, we devise a technique to automatically discover semantically enriched conjunctive relations in a KB. Our approach is generalized for any KB and requires no additional parameter tuning. Particularly, we employ classical rule mining techniques to perform relation composition on knowledge graphs to learn first order rules. We evaluate our proposed methodology on two state of the art information extraction systems, DBpedia and Yago with promising results in terms of generating high precision rules. Furthermore, we make the rules publicly available for community usage.

Keywords: Relation composition, Knowledge Bases, Rule mining, Ontologies, Information Extraction

1 Introduction

The Semantic Web research has gained a lot of prominence in the last few years. The large body of work in this area are in themselves an indicator for the potential of semantic representation. Every real world entity can have a semantic representation and is associated with a multitude of other entities with relationships. These relationships define association patterns between an entity pair. For instance, a statement like *Porsche was founded in Stuttgart* can be semantically represented by a directed relationship as `dbo:foundationPlace(db:Porsche, db:Stuttgart)`[1] Also note that, the semantic representation necessarily need not be via a direct relation, but can be equally stated with an inverse one. This

[1] We used the DBpedia [3,13] vocabulary here, but it can have any other semantics from other knowledge sources. The prefixes `dbo` refer to relation and concept namespace (http://dbpedia.org/ontology/) and `db` to instances/resources (http://dbpedia.org/resource/) in DBpedia. In the rest of the paper we omit these prefixes for brevity.

set of direct and/or inverse relations to/from a particular entity expresses explicit knowledge about the entity. But, often we miss out the more interesting, latent information, just by ignoring the relationship patterns beyond the explicit ones. For instance, exploring the entity relationships with other entities might unravel more information. For instance, `country(Stuttgart, Germany)` makes it easy to deduce that `Porsche` has an implicit connection with `Germany`. Such a path connecting a source entity (`Porsche`) with a target one (`Germany`) is called a relation path. And the number of transitions we execute from the source to the target is defined as hops. In particular, this is a 2-Hop scenario, schematically represented as ($\circ \xrightarrow{foundationPlace} \circ \xrightarrow{country} \circ$).

Now, we have two issues. First, a relation path of length n may not be semantically correct. An arbitrary conjunction of relations might not be useful or be the best choice. Second, if we find a semantically valid relation path, there is a degree of uncertainty associated with their authenticity. This allows us to formally define our problem statement. Let p_i be an arbitrary relation in some KB, K. If we have a conjunctive relation path of length n comprising of n such KB relations, then we want to find out if there is another relation h expressing the similar semantics as the path itself. This is referred to as a semantic relation composition, keeping close parity with role composition in ontology. Formally, $(p_1,.., p_i,.., p_n) \in K$, $\exists\ h \in K$,

$$conf : p_1 \wedge \cdots \wedge p_i \wedge \cdots \wedge p_n \equiv h \qquad (1)$$

In the rest of the paper we refer to this as a n-Hop rule. These rules are annotated with $conf$ [0, 1], the confidence score for the rules which indicates the degree of belief that h is indeed the representative of the conjunctive relation path. Intuitively, the primary goal of this work is to find a simplified representation of conjunction of n KB relations. In this context, we envision the applicability of our contribution in the following use cases:

Semantic Enrichment of Relations Each relation sequence in the body of a rule ($p_1 \wedge \cdots \wedge p_n$ in Expression 1 above) represents an obscure connection between the entities. While the head relation (h in Expression 1) of the rule is an hypothesis that it *might* be a simplified representation of the conjunction. In other words, we find out an alternate (shorter) path of traversing from the source entity to the target. Hence the name, relation composition. The rational is if we can find such paths, then they are semantically equivalent as they connect the same source-target pair of entities.

Detection of Transitive Relations: Rules composed of recurring relations like $p \wedge \cdots \wedge p \Rightarrow p$ can be exploited to extract candidate transitive relations. This schema detection is possible due to the structure of the derived rules. Especially if we observe this rule with a varying number of relations in the body, the evidence of the relation p being transitive is much higher.

Amendment of the KB with Missing Facts: The extracted rules can be used to generate facts missing in the KB by discovering links between entities. Our target is to generate composed rules with high confidence. These rules can be used as templates to deduce new KB facts. For instance, if we know with con-

siderable accuracy that `headquarter(a,b)` $\wedge$ `place(b,c)` $\Rightarrow$ `location(a,c)`, then a set of entities $a, b, c \in K$ satisfying the rule body can be used to deduced that `location(a,c)`. The higher the confidence of such a rule, higher is the confidence that the inferred fact is true. However, the higher the confidence, lesser missing links can be inferred.
Query Join Improvement: One of the major contribution of this work can be considered for semantic query performance improvements. Using composed relations instead of longer chains of relation paths in a query allows lesser number of joins to be performed. For instance, in terms of SPARQL, the query *select ?b where {db:IBM dbo:location ?b}* will have a faster response time than *select ?b where {db:IBM dbo:headquarter ?b. ?b dbo:place ?c}*.

In this paper, we make the following contributions, (i) proposing a KB independent, generalized and parameter tuning free approach to extract composed form of a set of conjunctive relations (ii) providing a document and ranked list of confidence annotated rules for the community. In the subsequent Section 2 we detail our methodology to find relations which can be composed in a given KB. We report on our evaluation schemes and experimental setup in Section 3. We highlight some of the closely related works in Section 4 and finally conclude in Section 5 with some possible scopes of extending this work.

2 Methodology

2.1 Rule Extraction

In this section, we focus on the details of semantic relation composition by extracting first-order logic rules from a KB. Since, a KB is essentially a collection of facts, we provide an excerpt from DBpedia consisting of three facts:

$$\begin{aligned}&\texttt{dbo:deathPlace(db:Marek\_Edelman,db:Warsaw)}\\&\texttt{dbo:battle(db:Marek\_Edelman,db:Warsaw\_Uprising)}\\&\texttt{dbo:place(db:Warsaw\_Uprising,db:Warsaw)}\end{aligned}\tag{2}$$

These sample facts are depicted as an knowledge graph in Fig. 1. As observed, relation `deathPlace` links `Marek_Edelman` with `Warsaw` directly. However, the knowledge graph illustrates a second way of reaching `Warsaw` starting from `Marek_Edelman`: The first hop via relation `battle` ends in `Warsaw_Uprising` and a subsequent hop from `Warsaw_Uprising` via relation `place` finally ends in `Warsaw`. This 2-Hop pattern connecting `Marek_Edelman` with `Warsaw` can be represented as a general logical conjunction of the individual relation hops as predicates and their respective entities[2] formulated in Equation 3:

$$\texttt{battle(ME,WU)} \wedge \texttt{place(WU,W)}\tag{3}$$

[2] For better legibility, `Marek_Edelman` is abbreviated with ME, `Warsaw_Uprising` with WU and `Warsaw` with W

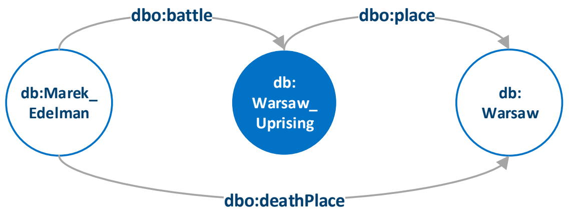

Fig. 1: Knowledge graph representing the introduced KB excerpt

This is a 2-Hop relation path, since the first entity can reach the third entity by applying two hops. This reveals that such a 2-Hop path in the KB from `Marek_Edelman` to `Warsaw` can be transformed to a first-order logic representation by introducing the direct path via `deathPlace`:

$$\texttt{battle(ME,WU)} \land \texttt{place(WU,W)} \Rightarrow \texttt{deathPlace(ME,W)} \tag{4}$$

In this rule the logical conjunction of the 2-Hop path is the body of the rule and the relation `deathPlace` is the head of the rule.

In this example `Warsaw_Uprising` serves as the join entity enabling the semantic composition as it appears in the range of the first relation and in the domain of the subsequent relation. A transformation of this rule containing explicit entities of our KB to a more general rule can be achieved by replacing all literals of the predicates with variables e_i:

$$\texttt{battle}(e_0, e_1) \land \texttt{place}(e_1, e_2) \Rightarrow \texttt{deathPlace}(e_0, e_2) \tag{5}$$

We should note that the rule above might still *appear* to be a candidate rule if `deathPlace` is replaced with `birthPlace`. We will discuss shortly that, we do not filter off any rule manually, rather we let the evidence based rules to determine its accuracy. The semantically weaker rule patterns are automatically weighted lesser than its stronger counterparts. With this simple example, we framed the notion of creating a first-order logic rule with semantic relation composition by exploiting conjunctive relation paths. We now consider a general knowledge graph structure (shown in Fig. 2) for formulating an abstract pattern for rules. As the head entity e_0 is linked to the tail entity e_n with n-Hops over relations $p_1, ..., p_n$ and both entities can be connected either directly by $p_{dir}(e_0, e_n)$ or inversely by $p_{inv}(e_n, e_0)$, the two general and formalized first-order logic rules can be extracted as :

$$\begin{aligned} p_1(e_0, e_1) \land p_2(e_1, e_2) \land ... \land p_n(e_{n-1}, e_n) \Rightarrow p_{dir}(e_0, e_n) \\ p_1(e_0, e_1) \land p_2(e_1, e_2) \land ... \land p_n(e_{n-1}, e_n) \Rightarrow p_{inv}(e_n, e_0) \end{aligned} \tag{6}$$

For rule extraction either one of these two patterns linking the head and tail entity can be used. These can be considered as templates for rule extraction. For a knowledge base, K and n relation hops between head and tail entity, the set

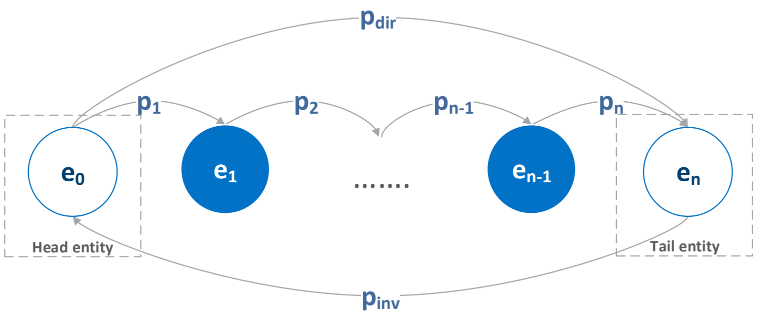

Fig. 2: n-Hop semantic relation composition from entity e_0 to entity e_n with $e_i, p_i, p_{dir}, p_{inv} \in K$

R^n_K contains all extractable n-Hop rules:

$$R^n_K = \{p_1 \wedge p_2 \wedge ... \wedge p_n \Rightarrow h \; | p_i, h \in K\} \tag{7}$$

Hence, the example rule in Equation 5 is a concrete element of the set $R^2_{DBPedia}$. Our goal is to generate these sets for varying values of n. In order to generate actual instances of such rules, we use SPARQL over public KB endpoints. The subsequent part describes a query template for a KB in RDF [10] with the SPARQL query language to extract KB structures depicted in fig. 2. Note that the following query is only pseudo-SPARQL.

```
PREFIX kb: <knowledgebase/>
SELECT ?p1 ?p2 ... ?pn ?pdir COUNT(?pdir)
WHERE {
        ?p1 a kb:prop ... ?pn a kb:prop . ?pdir a kb:prop.
        ?e0 ?p1 ?e1 .   ?e1 ?p2 ?e2 ... ?e[n-1] ?pn ?en.
        ?e0 ?pdir ?en
}
GROUP BY ?p1 ?p2 ... ?pn ?pdir
ORDER BY ?p1 ?p2 ... ?pn ?pdir
```

Listing 1.1: Main SPARQL query template for extracting n-Hop rules

In Listing 1.1, we define a prefix to refer to the current knowledge base. The $WHERE$ part describes the basic graph pattern that matches the structure in fig. 2. Hence, the relations $p_1, ..., p_n$ from the body of the rule are described by variables $?p_1, ..., ?p_n$ which are explicitly tagged with type *kb:prop*. The second part consisting of n triples of the form $?e_i$ $?p_{i+1}$ $?e_{i+1}$ formulates the join between the entities that are connected over the relations p_i in a sequential fashion. Finally the triple $?e_0$ $?p_{dir}$ $?e_n$ directly connects the head and tail entity with relation $?p_{dir}$. For additionally discovering rule patterns connecting the head entity e_0 with the tail e_n inversely, the previously described statement $?e_0$ $?p_{dir}$ $?e_n$ must be substituted with $?e_n$ $?p_{inv}$ $?e_0$ by just flipping head and tail entity. The

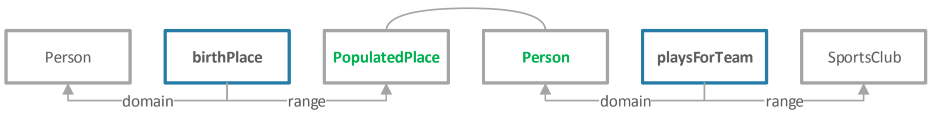

Fig. 3: Relation *birthPlace* and *playsForTeam* with their range and domain concepts

grouping of this graph by all the n-Hop path relations and the head relation allows to aggregate the instances of each rule and finally to select the number of instances which fulfill a specific rule. With varying n, we can generate specific rules satisfying Equation 6.

An easy way to extract rules from a KB for a specific relation list P is to build up queries with explicit relations for the n-Hop relation path instead of variables as in Listing 1.1 and only extract possible heads of a rule. This algorithm has a runtime of $\mathcal{O}(|P|^n)$, where n is the hop length. To speed up the semantic relation composition, the algorithm avoids querying the knowledge base for n-Hop relation combinations $p_1, ..., p_n$ which do not have joining concepts. It is evident from Equation 6, 4, that the conjunction of two relations is possible due to the join entity which co-occurs as object in the relation p_{i-1} but as subject in p_i. We use this feature to enhance our query time. We perform a simple check for $dom(p_i)$ and $ran(p_{i-1})$, which respectively maps relation p_i to its domain concept and relation p_{i-1} to its range concept. We present this in Figure 3, which shows the phenomena for a 2-Hop relation composition $\texttt{birthPlace}(e_0, e_1) \wedge \texttt{playsForTeam}(e_1, e_2)$ with their corresponding domain and range concepts. Obviously, the algorithm prunes this combination because $\texttt{PopulatedPlace} \not\sqsubseteq \texttt{Person}$. Since there is no entity which has the mutually exclusive ontological concepts `PopulatedPlace` and `Person` at the same time, no joining entity would be found enabling this relation composition and thus a pruning is appropriate.

2.2 Rule Cycles

In this semantic relation composition approach where direct relations are expressed with conjunctive relations paths, cycles between entities on the path potentially occur. Figure 4 depicts a general knowledge graph structure containing a cycle in the relation path from head to tail entity. Entities e_i and e_j are linked by $(j - i)$ relation hops and inversely connected over relation p_{cyc}. This knowledge graph cycle can be included arbitrary many times in the body of the rule. In order to avoid capturing cycles within the extracted rules, all entities $e_0, ..., e_n$ on the path must be pairwise distinct, $e_l \neq e_m$ for $l \neq m$. Subsequently, to ensure that this restriction is fulfilled and cycles are prohibited by excluding recurring entities, the following filter conditions are attached to the previously specified SPARQL query template:

```
FILTER(!regex(?e0, str(?e1)))
```

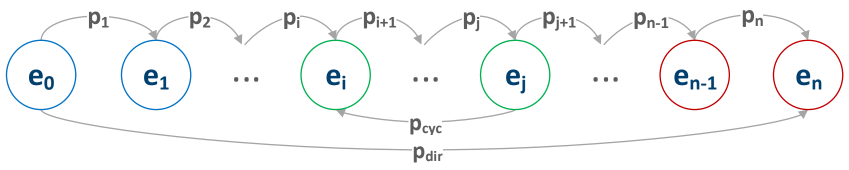

Fig. 4: General knowledge graph containing a cycle from entity e_i to e_j

```
....
FILTER(!regex(?e0, str(?en)))
FILTER(!regex(?e1, str(?e2)))
...
FILTER(!regex(?e1, str(?en)))
...
FILTER(!regex(?e[n-1], str(?en)))
```

Listing 1.2: SPARQL filters for the avoidance of cycles

2.3 Rule Metrics

In the introductory section, we mentioned two major sub problems. In this section, we discuss ways to quantify each of the first-order logic rules learnt from the KB. Often, there were scenarios where we observed very little evidence in support of some candidate semantic relation conjunctions. This motivated us to quantify each rule. As the first step, we gathered the evidences or the exact number of instances we observed the rule to be true. Each rule instance corresponds to a ground rule and counts as one evidence in favor of the general rule. Furthermore, to assign a normalized score to each rule, we employed association rule mining technique and metrics to this end. In particular, we compute the support and confidence [11,16] for each rule.

Obviously the straightforward way of learning a weight for a rule revealing the amount of evidence for this rule in the KB, is to count the number of $(n+1)$ valued tuple $(e_0, ..., e_n)$ evaluating the n-Hop rule $r \in R^n_K$ to true. Therefore, we formally define the following set $Inst_r$:

$$Inst_r = \{(e_0, ..., e_n) | r : p_1(e_0, e_1) \wedge p_2(e_1, e_2) \wedge ... \wedge p_n(e_{n-1}, e_n) \Rightarrow h\} \qquad (8)$$

An element of the instance set of our introducing 2-Hop example consists of three valued tuple (Marek_Edelman,Warsaw_Uprising,Warsaw) depicted in Fig.1 since it evaluates the rule in Equation 5 to true. The weight W_r of a rule r is defined as the number of instances fulfilling the rule, i.e. cardinality of $Inst_r$:

$$W_r = |Inst_r| \qquad (9)$$

The next rule metric we use is *support*, denoted by $supp(r)$, which specifies the number of instances of a rule r obtained among all rule instances of all rules in

#	Rule	RSL(r_k,r_i)
1	knowsPerson(e_0, e_1) $\wedge$ knowsPerson(e_1, e_2) $\Rightarrow$ knowsPerson(e_0, e_2)	0.8
2	isMarriedTo(e_0, e_1) $\wedge$ hasChild(e_1, e_2) $\Rightarrow$ hasChild(e_0, e_2)	0.65
3	affiliatedTo(e_0, e_1) $\wedge$ affiliatedTo(e_1, e_2) $\Rightarrow$ affiliatedTo(e_0, e_2)	0.44

Table 1: Example top-3 list for the RSL given that the Rule 13 (r_k) holds.

the KB. Therefore *supp*(r) allows to compare the weight of a rule r with all the other weights and essentially represents a relative weight. Formally,

$$supp(r) = \frac{W_r}{\sum_{i=1}^{|R^n_{KB}|} W_{r_i}} \tag{10}$$

And *confidence*, $conf(r)$, of a rule $r \in R^n_{KB}$ is the conditional probability that the head of the rule occurs under the condition that the body of the rule occurred already. The two functions $Body(r)$ and $Head(r)$ refer to the body and the head of the rule, respectively:

$$conf(r) = \frac{supp(Body(r) \cap Head(r))}{supp(Body(r))} \tag{11}$$

This implies that the more often entity triples (e_0, e_1, e_2) which fulfill the 2-Hop relation conjunction battle$(e_0, e_1) \wedge$ place(e_1, e_2) fulfill also the predicate deathPlace(e_0, e_2), the higher the confidence for the rule of Equation 5. We must note that the confidence is dependent on the number of evidences satisfying the rule or in a way the size of the KB. Our goal is to generate highly accurate rules and its composition and it is a rational choice to weigh such rules based on observed evidence. More often we see a rule pattern to hold true in a KB, more confident we are. In our situation, we necessarily do not consider other measures to consider rules which might be semantically correct but lack sufficient instances to prove their correctness.

The metrics introduced so far cater to individual rules only. But, we might be interested to know a set of rules likely to hold true under a given condition. We introduce the Rule Similarity Likelihood (RSL), a conditional probability indicating rule r_2 to hold given that rule r_1 holds. Formally,

$$RSL(r_1, r_2) = P(r_2|r_1) = \frac{|Inst_{r_1} \cap Inst_{r_2}|}{W_{r_1}} \tag{12}$$

This is also a measure of semantic similarity between two rules $r_1, r_2 \in R^n_K$. The corresponding instance sets $Inst_{r_1}$ and $Inst_{r_2}$ are compared by computing the overlapping amount with a conditional probability. Intuitively, the higher the number of rule instances satisfying r_1 and rule r_2, the higher the RSL for these rules. Table 1 illustrates an example top-3 list of the RSL given that the following rule, r_k, holds:

$$\texttt{knows}(e_0, e_1) \wedge \texttt{knows}(e_1, e_2) \Rightarrow \texttt{knows}(e_0, e_2) \tag{13}$$

3 Experiments

3.1 Datasets

For evaluation and experiments, we chose two state of the art knowledge bases.
Yago : A KB which extracts facts from Wikipedia. Moreover it integrates from other knowledge sources like the lexical database WordNet [15] and the geographical database GeoNames[3]. For our experiments we used only the $yagoFacts$ package from the CORE theme. This collection consists of about 5.5 Mio. facts build up by 36 relations.
DBpedia : A popular KB which also extracts structured information from Wikipedia. The extraction patterns therefore harvest facts from Wikipedia info boxes and exploit the underlying structure to generate high quality knowledge. We used the English version 3.9 consisting of about 580 Mio. facts and a filtered subset of 672 relations relevant for our rule extraction approach.
These two KBs are state of the art IE systems and considered large with respect to the information content. We choose these for their easy availability, wide usage and large number of relations. Furthermore, these are well structured KBs which allows queries over public endpoints (DBpedia) or data bases (Yago).
Rule Set: We have made the learnt rules publicly available for research use[4]. It is associated with a README file and rules for both DBpedia and Yago.

3.2 Annotation Scheme

For both the KBs we restrict our extraction and evaluation to 2-Hop and 3-Hop first-order logic rules. The top-k rules with descending confidence for each KB and for each $n = 2, 3$ were annotated manually, allowing a computation of the precision at k ($p@k$). In Expression 7 we defined the complete set of n-Hop rules. For the evaluation purpose, we decided to prune these sets based on a minimum weight. This means, from a selected set of weight thresholds of $W = 1, 100, 1000$, we select only those rules from R^n_K which have at least weight W. Any rule with a lower weight is rejected. Such a set is denoted as $R^n_{K,W}$.

For each experiment setting (different hops), three top-k lists are generated with the different weight thresholds. Top-k rule lists with a low W threshold usually have the top rules with high confidence values but low weights. These are often annotated as false since they are too specific to a particular context and cannot be considered a general rule pattern. The different weight thresholds enable a better analysis of the influence of the weight for the rule quality. For each top-k list we annotated the first 100 rules manually. Every one of those rules were presented to the annotator with additional example instances as evidences for the rule. Ones which were incorrect were marked so. The following rule is an example for an incorrect marked rule:

$$\texttt{canonizedPlace}(e_0, e_1) \wedge \texttt{city}(e_1, e_2) \Rightarrow \texttt{deathPlace}(e_0, e_2) \qquad (14)$$

[3] http://www.geonames.org/
[4] http://web.informatik.uni-mannheim.de/adutta/SRL.tar.gz

#	Rule	W	conf	supp
1	isMarriedTo(e_0, e_1) $\wedge$ hasChild(e_1, e_2) $\Rightarrow$ hasChild(e_0, e_2)	10782	0.57	0.0054
2	dealsWith(e_0, e_1) $\wedge$ isLocatedIn(e_1, e_2) $\Rightarrow$ isLocatedIn(e_0, e_2)	854	0.41	4.3E-4
3	dealsWith(e_0, e_1) $\wedge$ dealsWith(e_1, e_2) $\Rightarrow$ dealsWith(e_0, e_2)	3264	0.4	0.001
4	hasChild(e_0, e_1) $\wedge$ isPoliticianOf(e_1, e_2) $\Rightarrow$ isPoliticianOf(e_0, e_2)	417	0.36	2.1E-4
5	hasChild(e_0, e_1) $\wedge$ isAffiliatedTo(e_1, e_2) $\Rightarrow$ isAffiliatedTo(e_0, e_2)	578	0.36	2.9E-4

Table 2: Top-5 2-Hop rules from $R^2_{YAGO,100}$ excluding relation hasGender

#	Rule	W	conf	supp
1	asso.OfLocalGovernment(e_0, e_1) $\wedge$ country(e_1, e_2) $\Rightarrow$ country(e_0, e_2)	456	1.0	5.1E-5
2	daylightSavingTimeZone(e_0, e_1) $\wedge$ country(e_1, e_2) $\Rightarrow$ country(e_0, e_2)	157	1.0	1.7E-5
3	intercommunality(e_0, e_1) $\wedge$ country(e_1, e_2) $\Rightarrow$ country(e_0, e_2)	185	1.0	2.1E-5
4	federalState(e_0, e_1) $\wedge$ country(e_1, e_2) $\Rightarrow$ country(e_0, e_2)	4547	0.999	5.1E-4
5	arrondissement(e_0, e_1) $\wedge$ country(e_1, e_2) $\Rightarrow$ country(e_0, e_2)	3662	0.998	4.1E-4

Table 3: Top-5 2-Hop rules from $R^2_{DBPedia,100}$

3.3 Empirical Results

Table 2 shows the top-5 rules of $R^2_{YAGO,100}$. Notice that even though the first rule appears to be a very good rule in terms of its semantics, this top rule has only a confidence value of 0.57 and the confidence already dropped to 0.18 for the 15th rule. This rapid falling of the confidence values from the rules is continued to the 130th rule of the top-k list having only a confidence value of 1% left. Table 3 shows the top-5 rules of $R^2_{DBPedia,100}$. The confidence values are very high with a top confidence of 100% and decrease slowly in comparison with Yago 2-Hop rules. However, these top-5 rules have very similar semantics since all of them have the form $p(e_0, e_1) \wedge country(e_1, e_2) \Rightarrow country(e_0, e_2)$. Such a rule list is the expected outcome since spatial as well as *part-of* relationships are strongly present in a KB like DBpedia. Thus, these rules have not only high evidence in the KB expressed by high weights but also high confidence values. Most of the shown rules describe *part-of* relations for different regions in different countries like federal states, arrondissements, cantons and counties.

Figure 5 (I) shows the $p@k$ graphs for several top-k rule lists for $R^2_{YAGO,W}$ with the three different weight thresholds denoted by $w1, w100, w1000$. However, as visualized in the graph there are two different plots for $W = 1$, namely $w1.hasGender$ and $w1$, respectively. After the initial experiments of YAGO during the annotation, we noticed that the relation hasGender is contained in the resulting rules. For the following evaluation we discarded rules containing this relation from the several top-k lists. But to give an impression of the quality of the rule list including this property, we annotated one top-k list including hasGender with weight threshold $W = 1$ for comparative reasons. The top-k list with weight threshold $W = 1000$ has very high $p@k$ values for small k. This

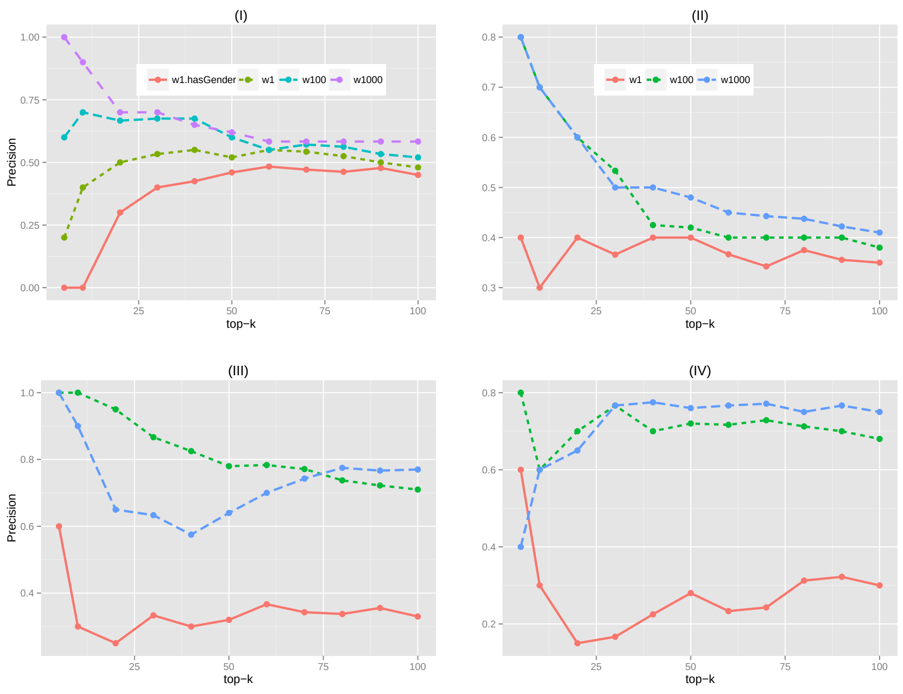

Fig. 5: Precision graphs for the rule sets with different weight thresholds. (I) R^2_{YAGO}. (II) R^3_{YAGO} (III) $R^2_{DBPedia}$, `dbo:` (IV) $R^2_{DBPedia}$, `dbp:`

is well-founded by the fact that the top rules have high confidence values plus high weights which produces more general and subsequently more true rules. The higher the weight threshold, the higher the precision of the rules. Figure 5 (III),(IV) shows the $p@k$ graphs for the three weight thresholds for $R^2_{DBPedia}$. In the data sets we differentiated between heads prefixed by `dbo`[5] and `dbp`[6]. Figure 5 (II) illustrates the $p@k$ for YAGO 3-Hop rules. In general, the precision for rules extracted from DBPEDIA is higher than for YAGO . For each rule set with a weight threshold W the number of extracted rules are illustrated in Table 4. In addition, the total number of rules with inverse heads is shown. The number of extracted 3-Hop rules from DBPEDIA is very low since we not finished analyzing DBPEDIA completely. For brevity, we omitted the remaining 3-Hop results but in general the precision is lower in comparison with that of 2-Hop rules.

[5] http://dbpedia.org/ontology/
[6] http://dbpedia.org/property/

	R^2_{YAGO}	R^3_{YAGO}	$R^2_{DBPedia}$	$R^3_{DBPedia}$
$W = 1$	744	2178	119951	434
$W = 100$	201	513	7189	37
$W = 1000$	66	167	1450	1
$W = 1(inv.)$	103	257	25949	37

Table 4: Number of extracted 2-Hop and 3-Hop rules from Yago and DBpedia

4 Related Work

Anyanwu et al. [2] propose an approach for the ranking of complex relationship search results on the Semantic Web called *SemRank*. The three presented Semantic Associations between two entities are composed of property sequences linking them semantically similarly to our n-Hop connection of head and tail entity. They introduce metrics to evaluate these different Semantic Associations holding between two entities of interest allowing finally an ordering of search results. The key feature of the ranking is the concept of predictability of a result which can be modulated and tuned by the user. The predictability is composed of the uniqueness of a result and the amount of deviance or refraction of a result path from the paths represented in the ontology schema.

A related work on discovering relation sequences between two entities of interest is the *RelFinder* proposed in [9,14]. This approach also focuses on exploiting extensive knowledge bases for revealing interesting semantic relationships between entities. Since exploring knowledge bases visually by knowledge graph representation can be very complicated on account of the vast number of entities and corresponding links between them, they propose an approach which simplifies the exploration process. Similarly to the proposed notion in this paper, they not only seek for direct relations between the two entities of interest but also for multiple hop relations.

Lao et al. [12] introduce the Path Ranking Algorithm (PRA) for probabilistic inference in large KBs. This algorithm enables a generation and prediction of new facts by executing random walks on the knowledge graph. PRA ranks graph nodes relative to a query node representing the probability that these two nodes are in fact semantically associated. The ranking of a single node is achieved by combining the rankings of the node from different paths in a linear model where each path feature is weighted.

Fleischhacker et al. [8] devise a technique to automatically reveal underlying axioms of relations by applying association rule mining on RDF data. This approach is able to detect property axioms like subsumption and disjointness, transitivity, symmetry and functionality based on confident association rules. The property axioms are generated by creating transaction tables containing the actual instances and their respective relations. For example, to detect symmetric properties the transaction table contains information about direct and inverse relations holding between entities simultaneously. These transaction tables are mined for rules which are eventually converted to actual property axioms.

Another approach described by Abedjan et al. [1] focuses on enhancement and enrichment of RDF data by employing association rule mining similarly to our approach. This rule-based approach enables predicate suggestion supporting users when creating new data entries, enrichment and amendment of the KB with missing facts and ontology schema improvement. Since the foundation for rule mining are facts in the SPO triples representation, they achieve the extraction of rules for different use cases by changing the context and target of mining. For example, in one configuration they mine subjects in the context of predicates. The resulting association rules hold between subjects with same predicates like `Merkel` $\rightarrow$ `Obama` since they are both persons and politicians and enable a clustering of similar subjects.

5 Conclusion & Future Work

Our empirical results prove that our proposed approach is in general able to extract semantically consistent rules from a knowledge base. This approach is strong in identifying typical relation hierarchies in knowledge bases. However, by just considering the confidence without including the weight into the examination, the quality of the rules is generally low. Thus, discovering good semantic relation compositions from a KB is accomplished by incorporating high thresholds for rule weights and confidence values. An appropriate weight threshold strongly depends on the used knowledge base but as empirically seen, for $W \geq 100$ we achieved good results. Furthermore, the quality of 2-Hop rules is higher than the quality of 3-Hop rules. For growing n we therefore assume a rapid drop in the quality of the rules, due to greater inclusion of noisy rules.

A major future work would be to extend the approach for other KBs. Therefore, experiments with the community created Knowledge Graph in FREEBASE [4] will be introduced to provide more variety of experimental rule sets and to adapt the implementation to MQL[7]. For enriching our rule extraction, we also consider other knowledge sources like ConceptNet[8] and Cyc[9]. Since YAGO, DBPEDIA and FREEBASE are structured IE systems, we would also apply our methodology on unstructured sources, which focus on probabilistic KBs, typically Open Information Extraction (OIE) [6] systems (NELL [5], REVERB [7]). We expect to face more challenges with OIE, since the entities are often not typed but free form texts, hence, entity linking and disambiguation would be an initial solution.

[7] https://developers.google.com/freebase/v1/mql-overview
[8] http://conceptnet5.media.mit.edu
[9] http://www.cycfoundation.org

References

1. Ziawasch Abedjan and Felix Naumann. Improving rdf data through association rule mining. *Datenbank-Spektrum*, 13(2):111–120, 2013.
2. Kemafor Anyanwu, Angela Maduko, and Amit Sheth. Semrank: ranking complex relationship search results on the semantic web. In *Proceedings of the 14th international conference on World Wide Web*, pages 117–127. ACM, 2005.
3. Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. *Dbpedia: A nucleus for a web of open data*. Springer, 2007.
4. Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.
5. Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.
6. Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. Open information extraction: The second generation. In *IJCAI*, volume 11, pages 3–10, 2011.
7. Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics, 2011.
8. Daniel Fleischhacker, Johanna Völker, and Heiner Stuckenschmidt. Mining rdf data for property axioms. In *On the Move to Meaningful Internet Systems: OTM 2012*, pages 718–735. Springer, 2012.
9. Philipp Heim, Sebastian Hellmann, Jens Lehmann, Steffen Lohmann, and Timo Stegemann. Relfinder: Revealing relationships in rdf knowledge bases. In *Semantic Multimedia*, pages 182–187. Springer, 2009.
10. Graham Klyne and Jeremy J Carroll. Resource description framework (rdf): Concepts and abstract syntax. 2006.
11. Kenneth Lai and Narciso Cerpa. Support vs. confidence in association rule algorithms. In *Proceedings of the OPTIMA Conference, Curicó*, 2001.
12. Ni Lao, Tom Mitchell, and William W Cohen. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 529–539. Association for Computational Linguistics, 2011.
13. Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 2014.
14. Jens Lehmann, Jörg Schüppel, and Sören Auer. Discovering unknown connections-the dbpedia relationship finder. *CSSW*, 113:99–110, 2007.
15. George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
16. Pang-Ning Tan and Vipin Kumar. Chapter 6. association analysis: Basic concepts and algorithms. *Introduction to Data Mining. Addison-Wesley. ISBN*, 321321367, 2005.