Amending RDF Entities with New Facts

Ziawasch Abedjan Felix Naumann

Hasso Plattner Institute, Potsdam, Germany
{ziawasch.abedjan,felix.naumann}@hpi.uni-potsdam.de

Abstract. Linked and other Open Data poses new challenges and opportunities for the data mining community. Unfortunately, the large volume and great heterogeneity of available open data requires significant integration steps before it can be used in applications. A promising technique to explore such data is the use of association rule mining. We introduce two algorithms for enriching RDF data. The first application is a suggestion engine that is based on mining RDF predicates and supports manual statement creation by suggesting new predicates for a given entity. The second application is knowledge creation: Based on mining both predicates and objects, we are able to generate entirely new statements for a given data set without any external resources.

1 Introduction

In the context of Linked Open Data (LOD), knowledge bases are usually incomplete or ontological structures are simply not available. Data inconsistencies and misusage of ontology axioms make it nearly impossible to infer new knowledge based on given axioms [6]. It is vital to achieve consistency within knowledge bases on the one hand by re-engineering ontology definitions [1] and to support the process of knowledge creation through value suggestions and auto-completion.

When creating new triples manually, one would hope that the creator exactly knows which properties and values should be created. However, regarding existing LOD data sets this is apparently not true. For instance, authors of Wikipedia infoboxes are often inexperienced and only infrequently edit such data. Such users might forget to use certain predicates or might use similar but not common predicates for a new entry (e.g., city instead of locationCity). Those heterogeneous entries make integration of the complete dataset difficult. Furthermore, a new user might be grateful for reasonable hints for creating a new entry. *Predicate suggestion* remedies the problem, providing users with a list of commonly used predicates. In case of Wikipedia infoboxes one could imagine to use the appropriate infobox-template for suggestions. However, reality is too complex to be covered by fixed static templates, and schema drift occurs[1] [1]. So,

[1] While the template Infobox company asks for a name, the vast majority of company infoboxes uses companyName instead.

an instance-based approach is able to suggest predicates based on existing entities. Extending statistical reasoning to object values, we can create an approach to *amend* datasets with completely new facts.

We propose an approach that applies association rule mining at RDF statement level by using the concept of *mining configurations* [2,3]. Our approach is complementary to traditional reasoning approaches, as we do not use ontology logics but simple basket analysis adapted to the triple structure of RDF data. The benefit of a mining approach is that outliers and individual faulty facts do not affect the overall performance as long as the occurrence of a specific incorrect fact is not statistically relevant. To this end, we make the following contributions in extension to [3]:

1. We elaborate the algorithms for schema and value suggestion for new knowledge base entries. Unlike related approaches we do not rely on external knowledge, such as ontologies or textual information.
2. We introduce a new approach for auto-amendment of RDF data with new triples, based on high confidence rules among objects.

2 Related Work

Association rule mining on RDF data is an emerging topic with several new use cases [1,2,4,11,17,20]. Nebot et al. present a shopping basket analysis framework in medical RDF data to discover drug and disease correlations among patients [17]. We introduced mining configurations [2,3], a methodology to generate association rules in different contexts of an RDF statement. We further introduced association-rule-based applications to reconcile ontologies with underlying data [1] and to discover synonyms in knowledge bases [4]. Following a statistical methodology, Völker and others presented an association rule based approach for schema induction [11,20], based on given class membership relations. Our approach generate facts beyond this explicitly given ontology information.

Most related work on mining the semantic web concentrates on inductive logic programming (ILP) and machine learning approaches [9,14,16]. ILP concentrates on mining answer-sets of queries towards a knowledge base. Based on a general reference concept, additional logical relations are considered to refine the entries in an answer-set. This approach assumes a clean ontological knowledge base, which is most often not available. ALEPH, WARMR [9], and Sherlock [19] are known systems to mine such rules. ALEPH is an ILP system based on Muggleton's Inverse Entailment Algorithm [16]. WARMR uses a declarative language to mine association rules on small sets of conjunctive queries. Sherlock uses a probabilistic graphical model to infer first order clauses from a set of facts for a given relation [19]. A recent system for association rule mining in RDF data is AMIE [12]. It concentrates on horn rules among relations, such as $hasChild(p,c) \wedge isCitizenOf(p,s) \rightarrow isCitizenOf(c,s)$. Based on support and confidence thresholds on the instantiations of the variable subject and objects, the rule generates new relations $isCitizenOf(c,s)$. In a number of experiments AMIE showed to be the most efficient and effective approach to generate new facts compared to ALEPH and WARMR [12]. Therefore, we experimentally compare our system to AMIE.

Complementing the ILP method, many machine learning systems, such as similarity-based class-membership predictions, kernel-based methods, and multivariate prediction models, have been introduced [18]. D'Amato et al. propose approaches to enrich ontologies by applying ILP to heterogeneous sources, such as RDBMS and web sources [7, 8]. Lisi et al. also present an approach to mine rules on ontologies and datalog programs [15]. Our approach does not rely on external data sources or structural information, such as ontologies or templates, but only the existing RDF statements of the current corpus.

3 Enriching RDF Data

To enrich RDF data, we distinguish two different scenarios: (1) *Suggestion* of predicates or object values for a given subject. (2) *Amendment* of RDF data with new triples. To this end, we only apply mining *configurations* in the context of subjects with predicates or objects as mining targets [2].

3.1 Association rules and triples

The concept of association rules has been widely studied in the context of market basket analysis [5], yet the formal definition is not restricted to any domain: Given a set of items $I = \{i_1, i_2, \ldots, i_m\}$, an association rule is an implication $X \rightarrow Y$ consisting of the *itemsets* $X, Y \subset I$ with $X \cap Y = \emptyset$. Given a set of transactions $T = \{t | t \subseteq I\}$, association rule mining aims at discovering rules holding two thresholds: minimum support and minimum confidence.

Support s of a rule $X \rightarrow Y$ denotes the fraction of transactions in T that include the union of the *antecedent* (left-hand side: itemset X) and *consequent* (right-hand side: itemset Y) of the rule, i.e., $s\%$ of the transactions in T contain $X \cup Y$. The confidence c of a rule denotes the statistical dependency of the *consequent* of a rule from the *antecedent*. The rule $X \rightarrow Y$ has confidence c if $c\%$ of the transactions T that contain X also contain Y.

To apply association rule mining to RDF data, it is necessary to identify the respective item set I as well as the transaction base T and its transactions. We follow the methodology of *mining configurations* [2], which is based on the subject-predicate-object (SPO) view of RDF data. Any part of the SPO statement can serve as a *context*, which is used for grouping one of the two remaining parts of the statement as the *target* for mining. So, a transaction is a set of target elements associated with one context element that represents the transaction id (TID). We call each of those *context* and *target* combinations a *configuration*.

3.2 Suggestion step

Suggestion of predicates or objects aims at two goals: First, a user authoring new facts for a certain subject might be grateful for reasonable hints. Second, system feedback might prevent the user from using inappropriate synonyms for predicates as well as objects. Because the suggestion workflow for both predicates and objects is identical, we describe our approach referring to predicates. The suggestion workflow requires two preprocessing steps:

1. Generate all association rules between predicates.
2. Create a *rule matrix*, which is a two dimensional predicate-predicate matrix, where one index identifies the antecedents and the other index identifies the consequents of a rule. Each entry specifies the confidence of the rule involving the specific antecedent and consequent. For missing rules the entry is zero by default.

When the user is inserting or editing facts related to a specific subject, the system is aware of all predicates that have already been inserted for the current subject. We denote the initial set of these predicates with ${}^{s}P_0$, where the raised s refers to the subject at hand. We use a raised letter to denote that a statement part is fixed by known values, e.g., ${}^{s}p^{o}$ denotes a predicate connecting the subject s with the object o. The following formula describes the set of predicates ${}^{s}P_0'$ out of the set of all predicates P that are to be suggested for the current subject s:

$${}^{s}P_0' = \{p \in P | \mathit{aConf}({}^{s}P_0, p) \geq \mathit{minConf}\}$$

${}^{s}P_0'$ contains all predicates from P, for which the function $aConf$ exceeds the minimum confidence threshold *minConf*. Here, *aConf* aggregates the confidence values of all available rules $\mathit{conf}(Q \rightarrow p)$ with $Q \subseteq {}^{s}P_0$ and creates one overall confidence value between 0 and 1. In our approach, we took the sum of all squared confidence values and normalized it by dividing by the number of schema elements in ${}^{s}P_0$:

$$aConf({}^{s}P_0, p) = \frac{\sum_{Q \in {}^{s}P_0} \mathit{conf}(Q \rightarrow p)^2}{|{}^{s}P_0|}$$

This choice ensures that the occurrence of few high confidence rules has more impact than many low confidence rules. Having computed the set ${}^{s}P_0'$, the results can be sorted by their aggregated confidence values and presented to the user. When the user chooses the next predicate p to insert into the data set, ${}^{s}P_1'$ has to be computed based on the new schema set ${}^{s}P_1 = {}^{s}P_0 \cup \{p\}$. Table 1 illustrates some SPO facts extracted from DBpedia. Now, imagine we are to insert a record for *D. Cameron* by beginning with the statements *"Cameron birthPlace London."* and *"Cameron orderInOffice Prime Minister."* Then ${}^{s}P_0 =$ $\{$*birthPlace, orderInOffice*$\}$ and the total set of remaining predicates would be $P = \{$*party, instrument*$\}$. Considering only rules of size 2, the set of predicate rules relevant for the next suggestion include *birthPlace* $\rightarrow$*party* with 66.7% confidence, *orderInOffice* $\rightarrow$ *party* with 100% confidence, and *birthPlace* $\rightarrow$ *instrument* with 33.3% confidence. Having $\mathit{minConf} = 50\%$, the predicate *party* would be added to P_0', because $\mathit{aConf}({}^{s}P_0, \mathit{party})$ is above *minConf*.

Table 1: Some SPO facts

Subject	**Predicate**	**Object**
Obama	birthPlace	Hawaii
Obama	party	Democrats
Obama	orderInOffice	President
Merkel	birthPlace	Hamburg
Merkel	orderInOffice	Chancellor
Merkel	party	CDU
Lennon	birthPlace	Liverpool
Lennon	instrument	Guitar

Suggesting objects is technically equivalent to that of predicates, but the number of distinct objects is by magnitudes larger, resulting in weaker rules. For instance, the DBpedia 3.6 data set contains 1,100 distinct predicates but 3,980,642 distinct objects. Furthermore, for a user, authoring an object value for a suggested predicate is more convenient than vice versa. For example, a user might have created the entry *B. Obama birthPlace Honolulu*. Following the object suggestion, the system might contain an object-to-object rule with enough confidence saying *Honolulu* $\rightarrow$ *USA* and suggests to add a new fact with *USA* as its object. The user might not know how the subject and the proposed object are connected and which predicate (birthPlace, residence, etc.) to choose. In addition to the semantical fitting of the predicate, the user has also to consider its appropriateness with regard to consistency among similar entities. Previously reported experiments also showed the significant superiority of predicate suggestion to object suggestions [3].

3.3 Amending with new statements

After we were able to suggest one missing part for a given subject, it is possible to also complete the remaining third part. For example, if the system decides to suggest the predicate *residence* for *B. Obama*, it is also able to choose the right object, e.g., *Washington D.C.* from the existing value range of *residence*. We call this method of creating new statements where the user decides which subject has to be amended with new triples *user-driven auto-amendment*. We described the user-driven auto-amendment in [3]. A different way of creating new statements is to let the system itself choose the subjects that should be amended with new triples. We call this approach *data-driven auto-amendment*.

In this data-driven approach the subject to be amended with a new fact is selected on the basis of existing high-confidence object rules. Our approach is based on the following intuitions:

1. For object rules $O' \rightarrow o$ with high confidence (above 90%) and $O' \subseteq O$, the subjects $S^{O'}$ occurring with the objects O' are also likely to occur with the object o. However, up to 10% of the subjects that occur with O' *violate* the rule by not occurring with o in any fact. Those facts may be absent, because of missing thoroughness during data creation.
2. A subject s should not be enriched with a fact containing object o if on the basis of the rules involving schema predicates ${}^{s}P$, no predicate can be chosen for the connection with o. This intuition allows a softening of the earlier intuition that expects all subjects that violate $O' \rightarrow o$ should be extended with a triple containing o.

One could adapt the intuitions based on high confidence rules also among predicates. However, the discovery of the appropriate object for a to-be-added predicate is much more cumbersome, because of the large number of available objects. Concerning the first intuition one could argue that some of these implicitly given facts can also be generated using ontological dependencies within the data. But, not all implicit dependencies in the real world are captured within an ontology. For example the high-confidence object rule *South Park* $\rightarrow$ *Trey Parker* among television episodes correctly suggests that Trey Parker is involved all episodes of South Park and should be added as the producer when absent. However, there can't be a general ontological rule that each episode of a series should have the same producer as listed for the complete series entity.

Algorithm overview. The algorithm for data-driven auto-amendment is divided into three steps:

1. Create predicate-predicate rule matrix as described in Sec. 3.2.
2. Generate high-confidence object rules $o_1, o_2, \dots o_n \rightarrow o$ using FP-Growth [13].
3. Create statements for subjects that violate high-confidence object rules: For each rule $o_1, o_2, \dots o_n \rightarrow o$ retrieve subjects S that violate the rule and for each $s \in S$ predict the predicate ${}^{s}p^{o}$ that connects s and the object o.

Statement creation. The third step is illustrated in Algorithm 1. For each object rule $O' \rightarrow o$ with $O' = o_1, o_2, \dots o_n$, all subjects s that occur with the antecedent of the rule but not with its consequent ($s \in S^{O'} - S^{o}$) are retrieved in line 2. This set contains all subjects that may be amended with new facts having the current object rule consequent o as their value. The choice of 90% as the high confidence threshold is arbitrary. We report evaluation results on this threshold in Sec. 4. As multiple object rules may contain the same consequent o, duplicate subject-object-pairs may be generated, which are naturally ignored. Further we exclude all rules $O' \rightarrow o$, with a more general rule $O'' \rightarrow o$, i.e., $O' \supset O''$, because $S^{O''} - S^{o}$ contains all subjects from $S^{O'} - S^{o}$. The rest of the algorithm is straightforward and starts with retrieving the candidate predicates P^{o} in line 5 and the schema predicates ${}^{s}P$ in line 6. The rating for each retrieved candidate predicate is computed in line 9. Given the set of schema elements ${}^{s}P$ and a candidate predicate $p \in P^{o}$, the confidence entries of the rule matrix are used to generate an overall rating for the specific candidate predicate p. The overall rating r_p for a candidate p is computed by $r_p = aConf({}^{s}P, p)$, the aggregated confidence of all rules with $Q \subseteq {}^{s}P$ as antecedent and p as consequent. After the candidate loop, the candidate with the highest rating is returned. Only if there is a predicate with a rating above a given threshold δ, e.g., 0 for any rating at all, the new fact *spo* consisting of the current subject s, the top rated predicate p, and current object rule consequent o is added to the set of new facts in line 14. Note, the number of new facts depends on the number of existent high-confidence rules and their corresponding set of violating subjects $S^{O'} - S^{o}$.

4 Experiments and Evaluation

To evaluate the accuracy and quality of our suggestion and auto-completion approaches we performed multiple experiments on multiple datasets. Table 2

Algorithm 1: Statement Generation Algorithm

```
   Data: objectRules /* with confidence above 90%*/
   Result: newStatements
1  foreach objectRule ∈ objectRules do
2      subjects ← getViolatingSubjects (objectRule);
3      consequentObject ← objectRule.getConsequent ();
4      foreach subject ∈ subjects do
5          candidates ← getCandidatePredicates (consequentObject);
6          schema ← getSchema (subject);
7          topRating ← 0;
8          foreach candidate ∈ candidates do
9              currentRating←getRating (schema, candidate);
10             if currentRating > topRating then
11                 topRating ← currentRating;
12                 predicate ← candidate;
13         if topRating > δ then
14             newStatements.add (subjects, predicate, consequentObject);
15 return newStatements
```

Table 2: Experimental data with distinct cardinalities

Data set	Triples	Subjects	Predicates	Objects
DBpedia 3.6	13,794,426	1,638,746	1,100	3,980,642
DBpedia 3.7	17,518,364	1,827,474	1,296	4,595,303
DBpedia 3.8	20,514,715	2,342,853	1,313	5,172,511
DBpedia 2.0[2]	7,034,868	1,376,877	10,321	1,778,459
YAGO2[2]	948,044	470,485	36	400,343
YAGO2s[2]	4,125,966	1,653,882	37	606,789

shows sizes of the different data sets. The entities in each data set correspond to one or more of the 250 existing types, and so we are able to perform experiments not only over all entities, but also more fine-grained on entities of a certain type, resembling data of specific domains. The last three datasets in Table 2 are cleaned knowledge bases provided by the authors of AMIE[2] [12].

We evaluate our amendment approach on multiple datasets. In particular, we adapt the scenario to compare the quality and efficiency of our approach with AMIE [12], using the implementation provided by the authors, and show that our system is competitive to AMIE achieving higher precision.

4.1 Comparing to AMIE

Our system as well as AMIE generates new facts based on evidence in knowledge bases. While we combine two mining configurations on statement level, AMIE mines horn rules between parameterized relations. We compared both systems with regard to prediction quality as well as efficiency.

[2] `http://www.mpi-inf.mpg.de/departments/ontologies/projects/amie/`

Table 3: Comparison to AMIE

Dataset	Approach	Facts	Hits	Precision
DBpedia 2.0	AMIE (63 rules)	2,359	55	2.3%
	mining conf.	2,335	146	**6.2%**
YAGO2	AMIE	1.658m	8.1k	0.5%
	mining conf.	2,086	52	**2.5%**
DBpedia 3.6	AMIE	-	-	-
	mining conf.	26,660	8.2k	**30.0%**

Prediction quality. We used the same datasets and evaluation scenario as AMIE for a fair comparison [12]. That means we ran both approaches on YAGO2 and DBpedia 2.0 and compared the predictions to YAGO2s and DBpedia 3.8, respectively. According to the original experiments reported by the developers, AMIE generates up to 74K hits in the YAGO2s dataset and 122K hits in DBpedia 3.8. However, the ratio of hits to the number of total predictions is below 1‰, as no confidence threshold was defined. To make a fair comparison we chose the best rules generated by AMIE, that contribute the same number of predictions as our approach. To this end, we sort the horn rules generated by AMIE by their PCA (partial completeness assumption) confidence as proposed by the authors and iterate the list in descending order. We configured our approach with 90% confidence threshold for object rules and 0.1% support for both object as well as predicate rules.

The results in Table 3 show that our approach leads to a higher precision. Of course the results confirm only that some facts are true, but cannot confirm that any of the generated facts are false unless checked by a human expert. On the YAGO2 dataset we report the precision for all produced rules by AMIE as the best rule already produced ten times more predictions than our approach. That specific rule generated for each relation *isMarriedTo(a,b)* the missing symmetric relation *isMarriedTo(b,a)* resulting in 4,424 hits in DBpedia 3.8. Due to memory consumption restrictions (50GB) we could not evaluate AMIE on the original DBpedia 3.6 dataset.

Resource consumption. AMIE is a multithreaded approach where the knowledge base is kept and indexed in main memory to compute support and PCA confidence values in appropriate time. The drawback is clearly the high memory consumption that requires up to 22 GB to discover rules on the DBpedia 2.0 dataset and 3.4 GB for the YAGO2 dataset. Our mining configuration system needs only two FP-Trees in memory (one for discovering predicate rules and one for object rules), resulting in less than 600 MB when running on DBpedia 2.0 and about 200MB for YAGO2. We perform both steps, mining predicates and objects, consecutively, which could just as well be done in parallel to improve runtime. The runtime of both approaches, AMIE as well as our mining configurations, on these datasets is under 1 minute.

In general, both approaches are valid strategies to amend a knowledge base with new facts. While AMIE generates new facts based on closed rules considering entire fact patterns as rule atoms, our approach is more granular in considering predicate correlations and object correlations independently.

Table 4: Generated statements on DBpedia v3.6 and their inclusion in v3.7

Type	Thing	Person	Album	Animal	Artist	Film	Organis.	Place	Species	Work
Facts	26,646	1,521	43	17,024	426	225	1,465	10,727	26,164	463
Included	8,237	278	25	8,753	219	27	187	1,140	9,448	67
Precision	30.9%	18.9%	58.1%	51.4%	51.5%	12%	18.1%	10.6%	36.1%	14.5%

4.2 Amendment quality on large datasets

To further analyze the capabilities of rule-based triple amendment we performed more experiments on the DBpedia 3.6 dataset. To identify strengths and weaknesses of the approach we also performed experiments on subsets of that dataset. Table 4 shows the number of generated facts and their inclusion ratio in the DBpedia 3.7 data set. The idea here is to automatically evaluate which percentage of generated triples is "validated" by a more up to date version of the same data source. The high precision of the results for Animals is caused by the fact that most of the newly added statements are Animal classification statements that have been missing in the older version because of the lack of thoroughness during data creation. Note, these classification statements do not correspond to the ontology class designators *rdfs:type*. Those statements were excluded to identify more interesting new facts.

While having 31% precision for $minconf = 90\%$, experiments on the complete data set (all entities of type *Thing*) with thresholds of 95% and 85% resulted into 44.3% precision having 5,866 new facts and 27.4% precision having 39,589 new facts, respectively. These results confirm our assumption that the higher this threshold is set the more precision can be achieved but the fewer facts may be generated. Those facts that were not included in DBpedia 3.7 are not necessarily wrong facts. We manually evaluated a random set of 50 not-included facts and achieved 72% precision.

How true is a high-confidence rule? Our intuition about high-confidence rules is that subjects that violate these rules are actually not intended to violate them. In other words, we assume that the number of those subjects that deliberately "violate" the rules is relatively low. We evaluated the quality of high-confidence rules $o_i \rightarrow o_j$ by manually verifying the relation of o_j to the violating subjects on 50 randomly selected violating subjects per data set.

Table 5 shows the results for four data sets (0.1% support and 90% confidence). Each data set corresponds to entities of the given type from DBpedia 3.6. We observe that the assumption holds for most objects rules in the domains *Person* and *Place*, such as *American Civil War* $\rightarrow$ *United States* for *Person* instances and *Vosges* $\rightarrow$ *Lorraine Region* for *Place* instances. High-confidence rules from movie data however are mostly the result of true exceptions.However, in movie data there are also interesting positive examples: the rule *Lon Chaney, Sr.* $\rightarrow$ *Silent Film* with 93% confidence is a rule where the violating movies are in fact silent movies and can be updated with the object *Silent Film*.

Note that our algorithm creates a new fact with the presumably missing object only if there is a predicate that matches the subject and the object.

Table 5: Percentage of true violations of a high-confidence rule

Type	Thing	Place	Person	Film
True Positives	37	41	42	22
Percentage	74%	82%	84%	44%

Completion with predicates. In previous experiments, we analyzed suggestion quality of predicates, where all top-10 recommendations had a success rate above 50% [3]. As the results conform to other recommendation scenarios, such as the experiments given in [10], we can conclude that association rule mining is a reasonable strategy for predicate suggestion. In the following, we analyze the predicate completion, which is similar to predicate suggestion with the difference that the object value is known. Given a subject s, its schema ${}^{s}P$, and a related object ${}^{s}o$, the aim of predicate completion is to select the most appropriate predicate ${}^{s}p^{o}$ out of all predicates P^{o} that have o in their range. We evaluated this step by applying the leave-one-out strategy: For each high-confidence object rule $o_i \rightarrow o_j$ we considered all subjects s^{o_j} that do not violate this rule and removed the connecting predicate ${}^{s}p^{o_j}$ between the subject s and the consequence object o_j and tried to predict ${}^{s}p^{o_j}$ based on the predicate matrix and the predicate candidates P^{o}. Table 6 illustrates the results for experiments on the complete data set (*type:Thing*) as well as the eight types with the most instances. In comparison to the suggestion evaluation, we see that the choice of the correct predicate is very accurate when knowing also the object of the statement. We achieve lower precision on *Person*, because many object rules there refer to locations, such as *Buenos Aires* $\rightarrow$ *Argentina*, and the predicate selection confuses predicates, such as *nationality*, *deathPlace*, and *birthPlace*. But even though the removed predicate is confused for these examples, the proposed predicate for the subject-object pair might still be a valid fact.

Table 6: Results for predicting removed predicates based on object rules

Type	Rules	Triples	Removed	Correct	Missing	Incorrect	Precision	Recall
Thing	189	13,794,426	1,019,785	919,815	1	99,969	90.2%	90.2%
Place	169	3,605,195	246,731	246,704	0	27	99.9%	99.9%
Person	37	3,618,525	30120	20440	0	9,680	67.9%	67.9%
Work	10	2,910,016	5725	5,669	0	56	99.0%	99.0%
Species	1,128	1,461,468	1,337,734	1,212,080	9	125645	90.6%	90.6%
Organisation	84	1,456,113	24,535	24,258	0	277	98.9%	98.9%
Animal	981	1,035,602	952,340	951,964	8	368	99.9%	99.9%
Album	6	934,005	782	683	0	99	87.3%	87.3%
Film	50	626,875	5,618	5,086	4	528	90.6%	90.5%

The column with the number of missing values represents the number of subject-object pairs, for which the algorithm does not select any predicate. For those pairs, the existing schema of the subject ${}^{s}P$ and the candidate predicates P^{o_j} are not related to each other. Because only few triples are concerned, the precision is always at least as high as the recall. One could assume that by increasing the minimum threshold for the selection decision (see Alg. 1, line 13) incorrect selections can be avoided by being marked as undecidable. However,

Table 7: Results for predicting 20,000 random predicates for each type

Type	Predictions	Correct	Missing	Incorrect	Precision	Recall
Thing	18,731	16,855	1,269	1,876	89.98%	84.28%
Place	19,775	18,359	225	1,416	92.84%	91.80%
Person	19,936	15,419	64	4,517	77.34%	77.10%
Work	19,865	17,291	135	2,574	87.04%	86.55%
Species	19,986	17,922	14	2,064	89.67%	89.61%
Organization	19,820	16,115	180	3,705	81.31%	80.58%
Animal	19,975	19,968	25	7	99.97%	99.84%
Album	19,861	19,121	239	640	96.27%	95.61%
Film	19,842	18,606	158	1,236	93.77%	93.03%

experiments showed that increasing the threshold yields more undecidable selections and fewer correctly selected predicates.

Finally, we evaluated the predicate selection based on randomly removed predicates. We wanted to examine whether the quality of the predicate selection depends on the choice of the objects and whether the fact that they are connected with consequences of high-confidence object rules influences the quality. Table 7 illustrates the results for predicting randomly removed predicates and shows that predicate selection does not depend on the choice of objects as there is no significant difference to the results in Tab. 6.

5 Conclusions

We showed how an association rule matrix can be used for suggesting both predicates as well as object values for a user who is inserting new statements for an entity. We proposed a user-driven and a data-driven approach for generating new facts without depending on external resources. We conclude that mining configurations is a reasonable approach to enrich RDF data. Comparing to state-of-the-art systems, we achieve higher precision allowing a manual verification step after generating new facts. In a real-world scenario, it is possible to drop object rules that denote weak hypotheses, as the predicate selection step works pretty accurate in turn. The generated facts and an online demonstration tool embedding our approach can be found on our website[3].

Further research includes joint reasoning on both RDF data using descriptive logic and statistical occurrences of statement parts, and reasoning and formalizing constraints and refinements that allow more complex configurations.

References

1. Z. Abedjan, J. Lorey, and F. Naumann. Reconciling ontologies and the web of data. In *CIKM*, pages 1532–1536, 2012.
2. Z. Abedjan and F. Naumann. Context and target configurations for mining RDF data. In *SMER*, pages 23–24, 2011.

[3] `http://www.hpi.uni-potsdam.de/naumann/projekte/mining_rdf_data.html`

3. Z. Abedjan and F. Naumann. Improving RDF data through association rule mining. *Datenbank-Spektrum*, 13(2):111–120, 2013.
4. Z. Abedjan and F. Naumann. Synonym analysis for predicate expansion. In P. Cimiano, O. Corcho, V. Presutti, L. Hollink, and S. Rudolph, editors, *The Semantic Web: Semantics and Big Data*, volume 7882 of *Lecture Notes in Computer Science*, pages 140–154. Springer Berlin Heidelberg, 2013.
5. R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, pages 207–216, 1993.
6. P. A. Bonatti, A. Hogan, A. Polleres, and L. Sauro. Robust and scalable linked data reasoning incorporating provenance and trust annotations. *Journal of Web Semantics*, 9(2), 2011.
7. C. d'Amato, V. Bryl, and L. Serafini. Semantic knowledge discovery from heterogeneous data sources. In A. Teije, J. Vlker, S. Handschuh, H. Stuckenschmidt, M. dAcquin, A. Nikolov, N. Aussenac-Gilles, and N. Hernandez, editors, *Knowledge Engineering and Knowledge Management*, volume 7603 of *Lecture Notes in Computer Science*, pages 26–31. Springer Berlin Heidelberg, 2012.
8. C. d'Amato, N. Fanizzi, and F. Esposito. Inductive learning for the semantic web: What does it buy? *Semantic Web Journal*, 1(1,2):53–59, Apr. 2010.
9. L. Dehaspe and H. Toivonen. Discovery of frequent datalog patterns. *Data Mining and Knowledge Discovery*, 3(1):7–36, Mar. 1999.
10. M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 22:143–177, 2004.
11. D. Fleischhacker, J. Völker, and H. Stuckenschmidt. Mining RDF data for property axioms. In *On the Move to Meaningful Internet Systems (OTM)*, volume 7566 of *Lecture Notes in Computer Science*, pages 718–735. Springer Berlin Heidelberg, 2012.
12. L. Galárraga, C. Teflioudi, K. Hose, and F. Suchanek. AMIE: Association rule mining under incomplete evidence in ontological knowledge bases. In *WWW*, 2013.
13. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*, pages 1–12, 2000.
14. J. Józefowska, A. Lawrynowicz, and T. Lukaszewski. The role of semantics in mining frequent patterns from knowledge bases in description logics with rules. *Theory Pract. Log. Program.*, 10:251–289, 2010.
15. F. A. Lisi and F. Esposito. Mining the semantic web: A logic-based methodology. In *Proceedings of the Int. Symposium on Foundations of Intelligent Systems (ISMIS)*, pages 102–111, Heidelberg, 2005.
16. S. Muggleton. Inverse Entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245–286, 1995.
17. V. Nebot and R. Berlanga. Mining association rules from semantic web data. In *Proceedings of the Int. Conference on Industrial Engineering and other Applications of applied Intelligent Systems (IEA/AIE)*, pages 504–513, Berlin, 2010.
18. A. Rettinger, U. Lösch, V. Tresp, C. d'Amato, and N. Fanizzi. Mining the semantic web - statistical learning for next generation knowledge bases. *Data Min. Knowl. Discov.*, 24(3):613–662, 2012.
19. S. Schoenmackers, O. Etzioni, D. S. Weld, and J. Davis. Learning first-order horn clauses from web text. In *EMNLP*, pages 1088–1098, 2010.
20. J. Völker and M. Niepert. Statistical schema induction. In G. Antoniou, M. Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, P. Leenheer, and J. Pan, editors, *The Semantic Web: Research and Applications*, volume 6643 of *Lecture Notes in Computer Science*, pages 124–138. Springer Berlin Heidelberg, 2011.