Evolutionary Discovery of Multi-Relational Association Rules from Ontological Knowledge Bases

Claudia d'Amato, Andrea G B Tettamanzi, Minh Tran Duc

Evolutionary Discovery of Multi-Relational Association Rules from Ontological Knowledge Bases

Claudia d'Amato[1], Andrea G. B. Tettamanzi[2], and Tran Duc Minh[2]

[1] University of Bari, Italy – E-mail: claudia.damato@uniba.it
[2] Université Côte d'Azur, Inria, CNRS, I3S, France
E-mail: andrea.tettamanzi@unice.fr, tdminh2110@yahoo.com

Abstract. In the Semantic Web context, OWL ontologies play the key role of domain conceptualizations while the corresponding assertional knowledge is given by the heterogeneous Web resources referring to them. However, being strongly decoupled, ontologies and assertional bases can be out of sync. In particular, an ontology may be incomplete, noisy, and sometimes inconsistent with the actual usage of its conceptual vocabulary in the assertions. Despite of such problematic situations, we aim at discovering hidden knowledge patterns from ontological knowledge bases, in the form of multi-relational association rules, by exploiting the evidence coming from the (evolving) assertional data. The final goal is to make use of such patterns for (semi-)automatically enriching/completing existing ontologies. An evolutionary search method applied to populated ontological knowledge bases is proposed for the purpose. The method is able to mine intensional and assertional knowledge by exploiting problem-aware genetic operators, echoing the refinement operators of inductive logic programming, and by taking intensional knowledge into account , which allows to restrict the search space and direct the evolutionary process. The discovered rules are represented in SWRL, so that they can be straightforwardly integrated within the ontology, thus enriching its expressive power and augmenting the assertional knowledge that can be derived from it. Discovered rules may also suggest new (schema) axioms to be added to the ontology. We performed experiments on publicly available ontologies, validating the performances of our approach and comparing them with the main state-of-the-art systems.

Keywords: Description Logics; Pattern Discovery; Evolutionary Algorithms

1 Introduction

The Semantic Web [3] is the new vision of the Web aiming at making Web contents machine readable besides of human readable. For the purpose, Web resources are semantically annotated with metadata referring to ontologies that are formal conceptualizations of domains of interest acting as shared vocabularies where the meaning of the annotations is formally defined. As such, annotated web resources represent the assertional knowledge given the intensional definitions provided with ontologies. Assertional and intensional ontological knowledge will be referred to as ontological knowledge base.

In the SW view, data, information, and knowledge are connected following best practices and exploiting standard Web technologies, e.g. HTTP, RDF and URIs. This allows to share and link information that can be read automatically by computers meanwhile creating a global space of resources semantically described.

The description of data/resources in terms of ontologies represents a key aspect in the SW. Interestingly, ontologies are also equipped with powerful deductive reasoning capabilities. However, due to the SW heterogeneous and distributed nature, ontological knowledge bases (KBs)[1] may turn out to be incomplete and noisy w.r.t. the domain of interest. Specifically, an ontology is incomplete when it is logically consistent (i.e., it contains no contradiction) but it lacks information (e.g., assertions, disjointness axioms, etc.) w.r.t. the domain of reference; while an ontology is noisy when it is logically consistent but it contains invalid information w.r.t. the reference domain. These situations may prevent the inference of relevant information or cause incorrect information to be derived.

However, by exploiting the evidence coming from the (assertional) knowledge, data mining techniques could be fruitfully exploited for discovering hidden knowledge patterns from ontological KBs, to be used for enriching an ontology both at terminological (schema) and assertional (facts) level, even in presence of incompleteness and/or noise. We present a method based on evolutionary algorithms, for discovering hidden knowledge patterns in the form of multi-relational association rules (ARs) coded in SWRL [14], which can be added to the ontology enriching its expressive power and increasing the assertional knowledge that can be derived. Additionally, discovered rules may suggest new axioms to be added to the ontology, such as transitivity and symmetry of a role, and/or concept/role inclusion axioms. Even if related work focussing on a similar goal can be found in the SW community (see [16, 15, 24, 11, 12]) and in the ILP community (see [21, 7, 19]), to the best of our knowledge, our work represents the first proposal that is able to discover hidden knowledge patterns in ontological knowledge bases while taking into account the background/ontological knowledge and exploiting the efficiency of genetic algorithms jointly with reasoning capabilities.

Evolutionary algorithms (EAs) [5, 9] are bio-inspired stochastic optimization algorithms, which exploit two principles that allow populations of organisms to adapt to their surrounding environment: genetic inheritance and survival of the fittest. Each individual of the population represents a point in the space of the potential solutions for the considered problem. The evolution is obtained by iteratively applying a small set of stochastic operators, known as *mutation*, *recombination*, and *selection*. Mutation randomly perturbs a candidate solution; recombination decomposes two distinct solutions and then randomly mixes their parts to form novel solutions; selection replicates the most successful solutions found in a population at a rate proportional to their relative quality. The resulting process tends to find, given enough time, globally optimal solutions to the problem much in the same way as in nature populations of organisms tend to adapt to their surrounding environment.

[1] By *ontological knowledge base*, we refer to a populated ontology, namely an ontology where both the schema and instance level are specified. The expression will be interchangeably used with the term ontology.

We build on these ideas and we combine them with recent work on relational ARs discovery from populated KBs in the Semantic Web [11, 4] to propose an EA for the discovery of mulit-relational ARs. The rationale for using EAs as a meta-heuristic for ILP is to mitigate the combinatorial explosion generated by the inductive learning of rich representations, such as those used in description logics [4], while maintaining the quality of the results.

Our solution is experimentally evaluated and comparisons with the main state-of-the art systems are provided. In the next section, basics are illustrated; the proposed method for discovering multi-relational association rules from ontological KBs is presented in Sect. 3 while its experimental evaluation is illustrated and discussed in Sect. 5. The main characteristics and value added of our proposal with respect to the state of the art are analyzed in Sect. 4. Conclusions and future work directions are drawn in Sect. 6.

2 Basics

We refer to ontological KBs described in Description Logics (DLs) [2] (being DLs the theoretical foundation of OWL), and we do not fix any specific DL. As usual in DLs, we will refer to a KB $\mathcal{K}$= ($\mathcal{T}, \mathcal{A}$) defined by means of the set of the terminological axioms, namely the TBox $\mathcal{T}$, and the set of assertional axioms, namely the ABox $\mathcal{A}$ and where the formal meaning of the axioms is given in terms of model-theoretic semantics. As for reasoning services, *instance checking*, which assesses if an individual is instance of a given concept, and *concept subsumption*, which consists in checking whether a concept (role) is subsumed by another concept (role), are exploited. Remind that DLs adopt the *open-world assumption* (OWA) which has consequences on answering to class-membership queries. Specifically, it may happen that an individual that cannot be proved to be instance of a certain concept is not necessarily a counterexample for it, rather it would be only interpreted as a case of insufficient (incomplete) knowledge for possibly proving the assertion. For more details concerning DLs see [2].

In the following the general definition of relational AR for an ontological KB $\mathcal{K}$ is given. Hence, the problem we want to address is formally defined.

Definition 1 (Relational Association Rule). *Given a populated ontological KB $\mathcal{K}$= ($\mathcal{T}, \mathcal{A}$), a* relational association rule *r for $\mathcal{K}$ is a Horn-like clause of the form: $body \rightarrow head$, where: (a) $body$ is a generalization of a set of assertions in $\mathcal{K}$ co-occurring together; (b) $head$ is a consequent that is induced from $\mathcal{K}$ and $body$*

Definition 2 (Problem Definition).

Given:

- *a populated ontological knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$;*
- *a minimum "frequency threshold", θ_f;*
- *a minimum "head coverage threshold", θ_{hc};*
- *a minimum "confidence improvement threshold", θ_{ic};*

Discover: *all frequent hidden patterns w.r.t θ_f, in the form of multi-relational ARs, that may induce new assertions for $\mathcal{K}$.*

Intuitively, a *frequent hidden pattern* is a generalization of a set of concept/role assertions co-occurring reasonably often (w.r.t. a fixed frequency threshold) together, showing an underlying form of correlation that is exploited for obtaining new assertions.

For representing the rules to be discovered (following Def. 2), the Semantic Web Rule Language (SWRL) [14] is adopted. It extends the set of OWL axioms of a given ontology with Horn-like rules.[2]

Definition 3 (SWRL Rule). *Given a KB $\mathcal{K}$, a SWRL rule is an implication between an antecedent (body) and a consequent (head) of the form:* $B_1 \wedge B_2 \wedge \ldots B_n \rightarrow H_1 \wedge \ldots \wedge H_m$*, where $B_1 \wedge \ldots \wedge B_n$ is the rule body and $H_1 \wedge \ldots \wedge H_m$ is the rule head. Each $B_1, \ldots, B_n, H_1, \ldots H_m$ is called an* atom*.*

An atom *is a unary or binary predicate of the form $P_c(s)$, $P_r(s_1, s_2)$,* $\mathsf{sameAs}(s_1, s_2)$ *or* $\mathsf{differentFrom}(s_1, s_2)$*, where the predicate symbol P_c is a concept name in $\mathcal{K}$, P_r is a role name in $\mathcal{K}$, s, s_1, s_2 are* terms*. A* term *is either a variable (denoted by x, y, z) or a constant (denoted by a, b, c) standing for an individual name or data value.*

The discovered rules can be generally called *multi-relational* rules since multiple binary predicates $P_r(s_1, s_2)$ with different role names of $\mathcal{K}$ could appear in a rule. The intended meaning of a rule is: whenever the conditions in the antecedent hold, the conditions in the consequent must also hold. A rule having more than one atom in the head can be equivalently transformed, due to the *safety condition* (see Def. 4), into multiple rules, each one having the same body and a single atom in the head. We will consider, w.l.o.g., only SWRL rules (hereafter just "rules") with one atom in the head.

2.1 Fixing the Language Bias

In this section, the adopted *language bias* is specified. It consists of a set of constraints giving a tight specification of the patterns worth considering, thus allowing to reduce the search space. We manage rules having only atomic concepts and/or role names of $\mathcal{K}$ as predicate symbols, and individual names as constants. Only *connected* [11] and non-redundant [15] rules satisfying the *safety condition* [13] are considered. Additionally, to guarantee decidability, only *DL-safe rules* are managed [17], that is rules interpreted under the DL-safety condition consisting in binding all variables in a rule only to explicitly named individuals in $\mathcal{K}$.[3] In the following, notations and formal definitions for the listed properties are reported.

Given an atom A, let $T(A)$ denote the set of all the terms occurring in A and let $V(A) \subseteq T(A)$ denote the set of all the variables occurring in A e.g. $V(C(x)) = \{x\}$ and $V(R(x, y)) = \{x, y\}$. Such notation may be extended to rules straightforwardly.

Definition 4 (Safety Condition). *Given a KB $\mathcal{K}$ and a rule $r = B_1 \wedge B_2 \wedge \ldots B_n \rightarrow H$, r satisfies the* safety condition *if all variables appearing in the rule head also appear in the rule body; formally if: $V(H) \subseteq \bigcup_{i=1}^{n} V(B_i)$,*

[2] The results is a KB with an enriched expressive power. More complex relationships than subsumption can be expressed. For details see [13].

[3] When added to an ontology, DL-safe rules are decidable and generate sound results but not necessarily complete.

Definition 5 (Connected Rule). *Given a KB $\mathcal{K}$ and a rule $r = B_1 \wedge B_2 \wedge \ldots B_n \rightarrow H$, r is* connected *iff every atom in r is transitively connected to every other atom in r.*

Two atoms B_i and B_j in r, with $i \neq j$, are connected *if they share at least a variable or a constant i.e. if $T(B_i) \cap T(B_j) \neq \emptyset$.*

Two atoms B_1 and B_k in r are transitively connected *if there exist in r, atoms $B_2, \ldots, B_{k-1}$, with $k \leq n$, such that, for all $i, j \in \{1, \ldots, k\}$ with $i \neq j$, $T(B_i) \cap T(B_j) \neq \emptyset$.*

Definition 6 (Non-redundant Rule). *Given a KB $\mathcal{K}$ and a rule $r = B_1 \wedge B_2 \wedge \ldots B_n \rightarrow H$, r is a non-redundant rule if no atom in r is entailed by other atoms in r wrt $\mathcal{K}$, i.e., if, $\forall i \in \{0, 1, \ldots, n\}$, with $B_0 = H$, results: $\bigwedge_{j \neq i} B_j \not\models_{\mathcal{K}} B_i$,*

Example 1 (Redundant Rule). Given $\mathcal{K}$ with $\mathcal{T} = \{\mathsf{Father} \sqsubseteq \mathsf{Parent}\}$ and the rule $r = \mathsf{Father}(x) \wedge \mathsf{Parent}(x) \rightarrow \mathsf{Human}(x)$ where Human is a primitive concept, r is redundant since the atom $\mathsf{Parent}(x)$ is entailed by the atom $\mathsf{Father}(x)$ wrt $\mathcal{K}$.

2.2 Metrics for Rule Evaluation

For determining the rules of interest for the goal in Def. 2, metrics for assessing the quality of a rule are necessary. In the following, the adopted metrics are summarized.

Given a rule $r = B_1 \wedge \ldots \wedge B_n \rightarrow H$, let us denote:

- $\Sigma_H(r)$ the set of distinct bindings of the variables occurring in the head of r, formally: $\Sigma_H(r) = \{binding\ V(H)\}$
- $E_H(r)$ the set of distinct bindings of the variables occurring in the head of r provided that the body and the head of r are satisfied, formally: $E_H(r) = \{binding\ V(H) \mid \exists\ binding\ V(B_1 \wedge \ldots \wedge B_n) : B_1 \wedge \ldots \wedge B_n \wedge H\}$. Since rules are connected, $V(H) \subseteq V(B_1 \wedge \ldots \wedge B_n)$
- $M_H(r)$ the set of distinct bindings of the variables occurring in the head of r also appearing as binding for the variables occurring in the body of r, formally: $M_H(r) = \{binding\ V(H) \mid \exists\ binding\ V(B_1 \wedge \ldots \wedge B_n) : B_1 \wedge \ldots \wedge B_n\}$

Following [11, 4], we recall a few basic definitions, modified from the classical ones (as given e.g. in [1]) to ensure monotonicity when atoms are added to the body of a rule.

Definition 7 (Rule Support). *Given a rule $r = B_1 \wedge \ldots \wedge B_n \rightarrow H$, its support is given by the number of distinct bindings of the variables in the head, formally:*

$$\mathrm{supp}(r) = |E_H(r)|. \tag{1}$$

Definition 8 (Head Coverage for a Rule). *Given a rule $r = B_1 \wedge \ldots \wedge B_n \rightarrow H$, its head coverage is given by the proportion of the distinct variable bindings from the head of the rule that are covered by the predictions of the rule:*

$$\mathrm{headCoverage}(r) = |E_H(r)|/|\Sigma_H(r)|. \tag{2}$$

Definition 9 (Rule Confidence). *Given a rule $r = B_1 \wedge \ldots \wedge B_n \rightarrow H$, its confidence is defined as the ratio of the number of distinct bindings of the predicting variables in the rule head and the number of their bindings in the rule body:*

$$\mathrm{conf}(r) = |E_H(r)|/|M_H(r)|. \tag{3}$$

An issue with these definitions, and particularly Def. 9, is that an implicit closed-world assumption is made, since no distinction between *false* predictions, i.e., bindings σ matching r such that $\mathcal{K} \models \neg H\sigma$, and *unknown* predictions, i.e., bindings σ matching r such that both $\mathcal{K} \models H\sigma$ and $\mathcal{K} \models \neg H\sigma$, is made. On the contrary, reasoning on ontologies is grounded on the open-world assumption. Additionally, our goal is to maximize correct predictions, not just describing the available data. To circumvent this limitation the following metric, generalizing the *PCA Confidence* [11], is introduced.

Definition 10 (Rule Precision). *Given a rule $r = B_1 \wedge \ldots \wedge B_n \rightarrow H$, its precision is given by the ratio of the number of correct predictions made by r and the total number of correct and incorrect predictions (predictions logically contradicting $\mathcal{K}$), leaving out the predictions with unknown truth value.*

This metric expresses the ability of a rule to perform correct predictions, but it is not able to take into account the induced knowledge, that is the *unknown* predictions. For this reason, the metrics proposed for this purpose in [10] are also considered for the evaluation in Sect. 5. They are recalled in the following:

- *match rate*: number of predicted assertions in agreement with facts in the complete ontology, out of all predictions;
- *commission error rate*: number of predicted assertions contradicting facts in the full ontology, out of all predictions;
- *induction rate*: number of predicted assertions that are not known (i.e., for which there is no information) in the complete ontology, out of all predictions.

3 Evolutionary Discovery of Relational Association Rules

Given a populated ontological KB, our goal is to discover frequent hidden patterns in the form of multi-relational ARs to be exploited for making predictions of new assertions in the KB. The discovered rules are DL-Safe and expressed in SWRL (see Sect. 2). Hence, they can be integrated with the existing ontology, resulting in a KB with an enriched expressive power [13, 14].

To achieve this goal, we propose to search the space of the SWRL rules that respect the language bias defined in Section 2.1 using an EA. The algorithm maintains a population of patterns (the individuals) and makes it evolve by iteratively applying a number of genetic operators. A pattern is the genotype of an individual and the corresponding rule is its phenotype. Since, like [11], our goal is to discover rules capable of making a large number of predictions, the fitness of a pattern is the head coverage (cf. Definition 8) of the rule constructed using the first atom of the pattern as the head and the remaining atoms as the body.

Algorithm 1 The CREATENEWPATTERN() Operator.

Input: a global variable A_f: a list of frequent atoms;
Output: r: a new, random pattern.
1: *length* $\sim \lceil \mathcal{U}(2, \text{MAX\_RULE\_LENGTH}) \rceil$
2: pick an atom $a \in A_f$ at random
3: $r \leftarrow a$
4: **while** r.SIZE() $<$ *length*$_{p'}$ **do**
5: $\quad r \leftarrow$ SPECIALIZE(r)
6: **return** r

The approach we propose may be regarded as alternative and complementary to level-wise generate-and-test algorithms for discovering relational ARs from RDF datasets [11] and, more specifically, recent proposals that take into account terminological axioms and deductive reasoning capabilities [4].

3.1 Representation

As in [11, 4], a pattern is represented as a list of atoms of the form $C(x)$ or $R(x, y)$, respecting the language bias, to be interpreted in conjunctive form. For each discovered frequent pattern, a multi-relational AR is constructed by considering the first atom in the list as the head of the rule and the remaining atoms as the rule body.

The genetic operators of initialization, recombination, and mutation, described in the following sections, are designed to enforce the language bias. An important consequence of the fact that patterns are intended to be transformed into rules for evaluation is that the order of atoms counts only insofar as one atom is in the head position (and, therefore, the head of the rule) or it is not (and, therefore, in the body of the rule). The relative position of atoms that are not in the head position is irrelevant.

3.2 Initialization

The initial population is seeded by n random patterns, randomly generated according to Alg. 1. This CREATENEWPATTERN() initialization operator requires a list A_f of frequent atoms, which is computed once and for all before launching the evolutionary process, and returns a new random pattern. A frequent atom is a pattern r consisting of a single atom of the form $C(x)$ or $R(x, y)$, such that $\text{supp}(r) \geq \theta_f$ (cf. Def. 7). A new pattern is seeded with a frequent pattern picked at random from A_f and a random target length between 2 and MAX_RULE_LENGTH is chosen; the specialization operator (detailed in Alg. 4), which adds a random atom to an existing pattern while respecting the language bias, is then called repeatedly, until the target length is attained.

3.3 Recombination

The recombination (or crossover) operator produces two offspring patterns from two parent patterns, by randomly exchanging their body atoms and fixing, if necessary, their variables so that they respect the language bias.

The operator, detailed in Algorithm 2, proceeds by creating a set L including all the atoms in the two input patterns and choosing a target length for the two offspring; then,

Algorithm 2 The Recombination Operator RECOMBINE(p, r).

Input: p, r: the two patterns to be recombined;
Output: p', r': two patterns that are a recombination of the input patterns.
$L \leftarrow p \cup r$
$\mathit{length}_{p'} \sim \lceil \mathcal{U}(2, \text{MAX\_RULE\_LENGTH}) \rceil$
$\mathit{length}_{r'} \sim \lceil \mathcal{U}(2, \text{MAX\_RULE\_LENGTH}) \rceil$
$p' \leftarrow \top$
while p'.SIZE() $<$ $\mathit{length}_{p'}$ **do**
    pick an atom $a \in L$ at random
    fix a so that $p' \wedge a$ respects the language bias
    $p' \leftarrow p' \wedge a$
$r' \leftarrow \top$
while r'.SIZE() $<$ $\mathit{length}_{r'}$ **do**
    pick an atom $a \in L$ at random
    fix a so that $r' \wedge a$ respects the language bias
    $r' \leftarrow r' \wedge a$
return p', r'

Algorithm 3 The Mutation Operator MUTATE(r).

Input: r: the pattern to be mutated;
Output: r': the mutated pattern.
if r.GETHEADCOVERAGE() $> \theta_{\text{mut}}$ **then**
    if r.SIZE() $<$ MAX_RULE_LENGTH **then**
        $r' \leftarrow$ SPECIALIZE(r)
    else
        $r' \leftarrow$ CREATENEWPATTERN()
else
    if r.SIZE() > 2 **then**
        $r' \leftarrow$ GENERALIZE(r)
    else
        $r' \leftarrow$ CREATENEWPATTERN()
return r'

atoms are picked from L at random and added to either pattern until the target length is attained, possibly changing their variables to ensure the language bias is respected.

Recombination is performed with probability p_{cross}.

3.4 Mutation

The mutation operator is based on the idea of specialization and generalization operators in inductive logic programming. Roughly speaking, a specialization operator appends a new atom to a pattern while preserving the language bias, whereas a generalization operator removes a body atom from a pattern while preserving the language bias.

Mutation is applied to every child pattern (resulting from recombination or not) with a small probability $p_{\text{mut}} \ll 1$.

Mutation, summarized in Algorithm 3, applies the specialization operator, if the head coverage of the rule corresponding to the pattern is above a given threshold θ_{mut}, or the generalization operator, if its head coverage is below θ_{mut}, to the pattern undergoing it.

The specialization operator is detailed in Algorithm 4. A specialization for a given pattern may be generated by applying one of the operators, defined in [4]:

Algorithm 4 The Specialization Operator SPECIALIZE().

Input: r: the pattern to be specialized;
Output: r': the specialized pattern.
1: $X \sim \mathcal{U}(0,1)$ {Extract a uniform random number from $[0,1)$}
2: **if** $X < \frac{1}{2}$ **then**
3: pick a concept name $C \in \mathcal{N}_C^{\text{freq}}$ at random
4: $r' \leftarrow$ ADDCONCEPTATOM(r, C)
5: **else**
6: pick a role name $R \in \mathcal{N}_R^{\text{freq}}$ at random
7: **if** $X < \frac{3}{4}$ **then**
8: $r' \leftarrow$ ADDROLEATOMWITHFRESHVAR(r, R)
9: **else**
10: $r' \leftarrow$ ADDROLEATOMWITHWITHALLVARSBOUND(r, R)
11: **return** r'

- ADDCONCEPTATOM, which adds an atom whose predicate symbol is a concept name in the ontology and its variable argument already appears in the pattern to be specialized. The predicate symbol can already appear in the pattern, in that case, a different variable name has to be used;
- ADDROLEATOMWITHFRESHVAR or WITHWITHALLVARSBOUND, which add an atom whose predicate symbol is a role name in the ontology and at least one of its variable arguments is shared with one or more atoms in the pattern while the other could be a shared or a new variable. The predicate symbol could be already existing in the pattern.

The operators are applied so that, at each step of the specialization process, rules in agreement with the language bias (see Sect. 2) are obtained. We refer the reader to [4] for a detailed description of these operators.

The generalization operator simply removes the last atom from a pattern. Given the way patterns are created and specialized, this guarantees that the resulting pattern respects the language bias.

3.5 Fitness and Selection

A pattern is evaluated by first constructing a rule from it, using the first atom of the pattern as its head and the remaining atoms as its body. Fitness is defined as the head coverage of the rule: $f(r) = \text{headCoverage}(r)$.

Selection is performed as in the breeder algorithm [20] by truncation with parameter τ: the n patterns in the population are sorted by decreasing fitness and the $\lfloor \tau n \rfloor$ fittest individuals are selected for reproduction. The remaining individuals are replaced by the offspring of the selected individuals.

3.6 Consistency Check

Inconsistent rules, i.e., rules that are unsatisfiable when considered jointly with the ontology, are of no use for knowledge base enrichment and have thus to be discarded.[4]

[4] As remarked in [15], the satisfiability check is useful only if disjointness axioms occur in the ontology. This check can be omitted (thus saving computational cost) if no disjointness axioms occur.

Algorithm 5 Evolutionary algorithm for the discovery of multi-relational ARs from a populated ontological KB.

Input: $\mathcal{K}$: ontological KB; θ_f: frequency threshold; θ_{hc}: *head coverage* threshold;
Output: *pop*: set of frequent patterns discovered from $\mathcal{K}$
Compute A_f, a list of frequent atoms in $\mathcal{K}$.
Initialize population *pop* of size n.
$g \leftarrow 0$
while $g <$ MAX_GENERATIONS **do**
  for $i = 0, 1, \ldots, n-1$ **do**
    compute fitness for *pop*$[i]$
  sort *pop* by decreasing fitness
  for $i = \lfloor \tau n \rfloor, \lfloor \tau n \rfloor + 2, \ldots, n-2$ **do**
    pop$[i] \leftarrow$ *pop*$[i$ **mod** $\lfloor \tau n \rfloor]$
    pop$[i+1] \leftarrow$ *pop*$[i+1$ **mod** $\lfloor \tau n \rfloor]$
    with probability p_{cross} **do** RECOMBINE(*pop*$[i]$, *pop*$[i+1]$)
    with probability p_{mut} **do** MUTATE(*pop*$[i]$)
    with probability p_{mut} **do** MUTATE(*pop*$[i+1]$)
  $g \leftarrow g + 1$
Remove redundant and inconsistent rules from the final population *pop*
return *pop*

Notice that this case should never occur if the ontological KB is consistent and noise-free. Nevertheless, since the proposed method can be also applied to noisy ontologies, it may happen that an unsatisfiable rule/pattern (when considered jointly with the ontology) is extracted, particularly if low *frequency* and *Head Coverage* thresholds (see Sect. 2.2 for details about the adopted metrics and related discussions) are considered.

Since checking rules for consistency may be very computationally expensive, we have decided not to check patterns for consistency during evolution. Instead, we defer this check and we apply it to the final population.

The satisfiability check is performed by calling an off-the-shelf OWL reasoner. Our current implementation is able to use two state-of-the-art OWL reasoners, namely Pellet [22] and Hermit [18]. However, we have observed that both reasoners fail to give an answer within a reasonable time for some patterns. This happens relatively seldom and not necessarily with the same patterns for either reasoner; however, given the large number of pattern our algorithm generates, these cases have a high chance of occurring in every run. As a workaround, we have introduced a time-out, which is an additional parameter of the algorithm, after which the reasoner is interrupted. When this happens, we discard the problematic pattern, since we have observed that, in general, patterns that take too long to be checked are either inconsistent or uninteresting.

The overall flow of the EA may be summarized as in Alg. 5. The parameters of the algorithm are summarized in Table 1.

The rules corresponding to the patterns returned by the EA are straightforwardly obtained and coded in SWRL by considering, for each pattern, the first atom as the head of the rule and the remaining as the rule body.

4 Related Work

The exploitation of data mining methods for discovering hidden knowledge patterns is not new in the SW context. First proposals have been formalized in [16, 15], where solutions for discovering frequent patterns in the form of, respectively, DATALOG clauses

Table 1. Parameters of the evolutionary algorithm.

Parameter	Description
n	Population size
MAX_GENERATIONS	Maximum number of generations
MAX_RULE_LENGTH	Maximum pattern length
p_{cross}	Crossover rate
p_{mut}	Mutation rate
θ_{mut}	Head coverage threshold for mutation
τ	Truncation proportion
T/O	Reasoner time-out

and conjunctive queries from hybrid sources of knowledge (i.e. a rule set and an ontology) have been presented. These methods are grounded on a notion of *key*, standing for the basic entity/attribute to be used for counting elements for building the frequent patterns. Unlike these methods, our solution focuses on an ontological KB and does not require any notion of *key* and as such it is able to discover any kind of frequently hidden knowledge patterns in the ontology. A method for learning ARs from RDF datasets, with the goal of inducing a schema ontology has been proposed in [24], while a method for inducing new assertional knowledge from RDF datasets has been presented in [11] and further optimized in [12]. Differently from our approach, these two methods do not take into account any background/ontological knowledge and do not exploit any reasoning capabilities. Furthermore, our solution allows to discover rules that can be directly added to the ontology, which is not the case for the existing methods.

As regards exploiting EAs in combinations with ILP, several started to appear in the literature at the beginning of the new millennium. An EA has been exploited as a wrapper around a population of ILP algorithms in [21]; alternatively, a hybrid approach combining an EA and ILP operators has been proposed in [8, 6, 7]. A similar idea is also followed by [23, 19], in which a genetic algorithm is used to evolve and recombine clauses generated by a stochastic bottom-up local search heuristic.

The rationale for using evolutionary algorithm as a meta-heuristic for ILP is to mitigate the combinatorial explosion generated by the inductive learning of rich representations, such as those used in description logics [4], while maintaining the quality of the results.

5 Experiments and Results

We tested our method on the same publicly available ontologies used in [4]: Financial,[5] describing the banking domain; Biological Pathways Exchange (BioPAX) Level 2 Ontology,[6] describing biological pathway data; and New Testament Names Ontology (NTN),[7] describing named things (people, places, and other classes) in the New Testament, as well as their attributes and relationships. Details on these ontologies are reported in Table 2.

[5] `http://www.cs.put.poznan.pl/alawrynowicz/financial.owl`.
[6] `http://www.biopax.org/release/biopax-level2.owl`.
[7] `http://www.semanticbible.com/ntn/ntn-view.html`.

Table 2. Key facts about the ontological KBs used.

Ontology	# Concepts	# Roles	# Indiv.	# Declared Assertions	# Decl.+Derived Assertions
Financial	59	16	1000	3359	3814
BioPAX	40	33	323	904	1671
NTMerged	47	27	695	4161	6863

The first goal of our experiments consisted in assessing the ability of the discovered rules to predict new assertional knowledge for a considered ontological KB. For that purpose, different samples of each ontology have been built for learning multi-relational ARs (as presented in Sect. 3) while the full ontology versions have been used as a test set. Specifically, for each ontology three samples have been built by randomly removing, respectively, 20%, 30%, and 40% of the concept assertions, according to a stratified sampling procedure.

We ran the evolutionary algorithm by repeating for each run the sampling procedure. We performed 10 runs for each ontology and parameter setting, using the following parameters setting: $n = 1000$, MAX_GENERATIONS $= 1000$, MAX_RULE_LENGTH $= 10$, $p_{\text{cross}} = 0.6$, $p_{\text{mut}} = 0.4$, $\theta_{\text{mut}} = 0.2$, $\tau = \frac{1}{5}$, $\theta_f = 1$, $\theta_{hc} = 0.01$, $\theta_{ic} = 0.001$. As for the reasoner time-out (T/O), after some preliminary tests, we concluded that 10 seconds were enough to reduce the number of discarded patterns to a minimum; nevertheless, in the experiments we have also considered time-outs of 20 and 30 seconds to be on the safe side. As a results, three sets of 10 runs were performed for each ontology, one for each combination of sample and time-out, and the final population of each run were filtered using three time-outs, yielding a total of nine sets of 10 results.

As in [11], we applied the discovered rules to the full ontology versions and collected all predictions, that is the head atoms of the instantiated rules. All predictions already contained in the reduced ontology versions were discarded while the remaining predicted facts were considered. A prediction is assessed as *correct* if it is contained/entailed by the full ontology version and as *incorrect* if it is inconsistent with the full ontology version. Results (see Table 3) have been averaged over the different runs for each parameter setting and have been measured in terms of: *precision* (see Def. 10), *match rate*, *commission error rate*, and *induction rate* (see Sect. 2.2).

These results fully confirm the capability of the proposed approach to discover accurate rules (precision = 1 on all samples of all ontologies considered) and, which is even more relevant, to come up with rules that induce previously unknown facts (induction rate > 0), with a very large absolute number of predictions by the standards of alternative state-of-the art approaches.

It is thus interesting to compare the performance of the proposed evolutionary method to those of the two state-of-the-art level-wise generate-and-test algorithms which are closest to it in purpose, namely the multi-relational association rule discovery (RARD) method proposed by d'Amato *et al.* [4] and AMIE [11]. The comparison is performed by considering the top m rules wrt. their match rate, with m equal to the number of rules discovered by AMIE, when few rules were discovered, or to 10 for the other cases. Averaged results are reported in Table 4, further corroborating the claim that the proposed

Table 3. Average ($\pm$ standard deviation) performance metrics on each ontology.

Ontology	Sample	T/O	Match Rate	Comm. Rate	Ind. Rate	Precision	Number of # Predictions
Financial	20%	10s	0.983 ± 0.017	0	0.017 ± 0.17	1.0	32,607 ± 39,099
		20s	0.983 ± 0.017	0	0.017 ± 0.17	1.0	32,607 ± 39,099
		30s	0.983 ± 0.017	0	0.017 ± 0.17	1.0	32,607 ± 39,099
	30%	10s	0.970 ± 0.034	0	0.030 ± 0.034	1.0	64,875 ± 60,514
		20s	0.970 ± 0.034	0	0.030 ± 0.034	1.0	64,875 ± 60,514
		30s	0.970 ± 0.034	0	0.030 ± 0.034	1.0	64,875 ± 60,514
	40%	10s	0.933 ± 0.105	0	0.067 ± 0.105	1.0	47,264 ± 49,700
		20s	0.933 ± 0.105	0	0.067 ± 0.105	1.0	47,264 ± 49,700
		30s	0.933 ± 0.105	0	0.067 ± 0.105	1.0	47,264 ± 49,700
BioPAX	20%	10s	0.808 ± 0.087	0	0.192 ± 0.087	1.0	21,065 ± 8,914
		20s	0.807 ± 0.085	0	0.193 ± 0.085	1.0	22,397 ± 8,737
		30s	0.807 ± 0.085	0	0.193 ± 0.085	1.0	22,397 ± 8,737
	30%	10s	0.877 ± 0.056	0	0.123 ± 0.056	1.0	19,697 ± 8,846
		20s	0.877 ± 0.056	0	0.123 ± 0.056	1.0	19,697 ± 8,847
		30s	0.877 ± 0.056	0	0.123 ± 0.056	1.0	19,697 ± 8,847
	40%	10s	0.877 ± 0.056	0	0.113 ± 0.056	1.0	19,621 ± 12,811
		20s	0.877 ± 0.056	0	0.113 ± 0.056	1.0	19,621 ± 12,811
		30s	0.877 ± 0.056	0	0.113 ± 0.056	1.0	19,621 ± 12,811
NTMerged	20%	10s	0.578 ± 0.118	0	0.422 ± 0.118	1.0	3,324,264 ± 891,161
		20s	0.572 ± 0.119	0	0.428 ± 0.119	1.0	3,702,706 ± 826,273
		30s	0.571 ± 0.119	0	0.429 ± 0.119	1.0	3,748,387 ± 827,350
	30%	10s	0.707 ± 0.080	0	0.293 ± 0.080	1.0	3,489,818 ± 1,089,094
		20s	0.705 ± 0.081	0	0.295 ± 0.081	1.0	3,781,877 ± 1,415,805
		30s	0.705 ± 0.081	0	0.295 ± 0.081	1.0	3,790,930 ± 1,408,588
	40%	10s	0.665 ± 0.131	0	0.335 ± 0.131	1.0	3,564,421 ± 1,290,532
		20s	0.664 ± 0.131	0	0.336 ± 0.131	1.0	3,643,770 ± 1,320,093
		30s	0.662 ± 0.131	0	0.338 ± 0.131	1.0	3,708,683 ± 1,363,246

evolutionary algorithm can substantially boost the performance of multi-relational AR discovery. The large number of predictions made, on average, by the rules discovered by the evolutionary algorithm, depends on our language bias, which allows open rules (such that $V(B) \setminus V(H) \neq \emptyset$): open rules may generate substantially larger number of predictions than closed rules.

6 Conclusions

We presented an evolutionary method for discovering multi-relational ARs, coded in SWRL, from ontological KBs, to be used primarily for enriching assertional knowledge. The proposed approach has been experimentally evaluated through its application to publicly available ontologies and compared to the two most relevant state-of-the-art algorithms having the same goal.

Table 4. Comparison of EA vs. RARD and AMIE w.r.t. the number of rules discovered.

Ontology	Samp.	T/O	# Rules			Top			
			EA	RARD	AMIE	m	# Predictions		
							EA	RARD	AMIE
Financial	20%	10s	94.1 ± 33.7	177	2	2	14,442 ± 17,280	29	208
		20s	94.1 ± 33.7				14,442 ± 17,280		
		30s	94.1 ± 33.7				14,442 ± 17,280		
	30%	10s	86 ± 32	181	2	2	29,890 ± 29,576	57	197
		20s	86 ± 32				29,890 ± 29,576		
		30s	86 ± 32				29,890 ± 29,576		
	40%	10s	78 ± 50	180	2	2	18,958 ± 21,954	85	184
		20s	78 ± 50				18,958 ± 21,954		
		30s	78 ± 50				18,958 ± 21,954		
BioPax	20%	10s	144.1 ± 46.2	298	8	8	1,902.3 ± 755.7	25	2
		20s	144.4 ± 46.7				2,045.6 ± 740.9		
		30s	144.4 ± 46.7				2,045.6 ± 740.9		
	30%	10s	188.2 ± 25.5	283	8	8	1,653.1 ± 779.1	34	2
		20s	188.2 ± 25.5				1,653.1 ± 779.1		
		30s	188.2 ± 25.5				1,653.1 ± 779.1		
	40%	10s	159.3 ± 37.7	272	0	8	1,704.4 ± 1,437	50	0
		20s	159.3 ± 37.7				1,704.4 ± 1,437		
		30s	159.3 ± 37.7				1,704.4 ± 1,437		
NTMerged	20%	10s	1,035.4 ± 588.7	243	1,129	10	85,457 ± 25,754	620	420
		20s	1,044.4 ± 592.8				97,622 ± 24,878		
		30s	1,045.9 ± 592.6				98,470 ± 25,261		
	30%	10s	942.4 ± 217.1	225	1,022	10	103,962 ± 32,449	623	281
		20s	945.6 ± 218				114,940 ± 41,960		
		30s	946.1 ± 218.4				11,940 ± 41,960		
	40%	10s	893.7 ± 473.5	239	1,063	10	101,102 ± 38,777	625	332
		20s	895.6 ± 473.9				102,569 ± 38,828		
		30s	897 ± 473.2				103,100.4 ± 38,903		

For the future, we intend to focus on two main aspects: 1) scalability, by considering experimenting our method on datasets from the Linked Data Cloud; 2) reducing the search space for discovering ARs by further exploiting the expressive power of the representation language by considering the presence of hierarchy of roles.

References

1. R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proc. of the Int. Conf. on Management of Data*, pages 207–216. ACM Press, 1993.
2. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge Univ. Press, 2003.

3. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 2001.
4. C. d'Amato, S. Staab, A. Tettamanzi, M. Tran, and F. Gandon. Ontology enrichment by discovering multi-relational association rules from ontological knowledge bases. In *Proc. of SAC 2016*. ACM, 2016.
5. K. DeJong. *Evolutionary Computation: A unified approach*. MIT Press, Cambridge, MA, 2002.
6. Federico Divina. *Hybrid Genetic Relational Search for Inductive Learning*. PhD thesis, Vrije Universiteit Amsterdam, 2004.
7. Federico Divina. *Genetic Relational Search for Inductive Concept Learning: A Memetic Algorithm for ILP*. LAP LAMBERT Academic Publishing, 2010.
8. Federico Divina and Elena Marchiori. Evolutionary concept learning. In William B. Langdon *et al.*, editor, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, New York, USA, 9-13 July 2002*, pages 343–350. Morgan Kaufmann, 2002.
9. A. Eiben and J. Smith. *Introduction to Evolutionary Computing*. Springer-Verlag, Berlin, 2003.
10. N. Fanizzi, C. d'Amato, and F. Esposito. Learning with kernels in description logics. In F. Železný and N. Lavrac, editors, *Proceedings of the 18th Int. Conf. on Inductive Logic Programming (ILP 2008)*, volume 5194 of *LNCS*, pages 210–225. Springer, 2008.
11. L. Galárraga, C. Teflioudi, K. Hose, and F. Suchanek. AMIE: Association rule mining under incomplete evidence in ontological knowledge bases. In *Proc. of the 22th International Conference on World Wide Web (WWW '13)*, pages 413–422. ACM, 2013.
12. L. Galárraga, C. Teflioudi, K. Hose, and F. Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *The VLDB Journal*, 24(6):707–730, December 2015.
13. I. Horrocks and P. F. Patel-Schneider. A proposal for an OWL rules language. In *Proc. of the Int. Conf. on World Wide Web*, pages 723–731. ACM, 2004.
14. I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML, 2004.
15. J. Józefowska, A. Lawrynowicz, and T. Lukaszewski. The role of semantics in mining frequent patterns from knowledge bases in description logics with rules. *Theory and Practice of Logic Programming*, 10(3):251–289, 2010.
16. F. A. Lisi. AL-QuIn: An onto-relational learning system for semantic web mining. *Int. J. of Semantic Web and Information Systems.*, 2011.
17. B. Motik, U. Sattler, and R. Studer. Query answering for OWL-DL with rules. *Web Semantics*, 3(1):41–60, 2005.
18. Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau reasoning for description logics. *Journal of Artificial Intelligence Research*, 36:165–228, 2009.
19. Stephen Muggleton and Alireza Tamaddoni-Nezhad. QG/GA: a stochastic search for progol. *Machine Learning*, 70(2-3):121–133, 2008.
20. H. Mühlenbein and D. Schlierkamp-Voosen. The science of breeding and its application to the breeder genetic algorithm (BGA). *Evolutionary Computation*, 1(4):335–360, Winter 1993.
21. Philip G. K. Reiser and Patricia J. Riddle. Scaling up inductive logic programming: An evolutionary wrapper approach. *Applied Intelligence*, 15(3):181–197, 2001.
22. Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics*, 5(2):51–53, June 2007.
23. Alireza Tamaddoni-Nezhad and Stephen Muggleton. A genetic algorithms approach to ILP. In Stan Matwin and Claude Sammut, editors, *Inductive Logic Programming, 12th International Conference, ILP 2002, Sydney, Australia, July 9–11, 2002. Revised Papers*, volume 2583 of *Lecture Notes in Computer Science*, pages 285–300. Springer, 2002.
24. J. Völker and M. Niepert. Statistical schema induction. In *ESWC'11 Proc.*, volume 6643 of *LNCS*, pages 124–138. Springer, 2011.