Exception-Enriched Rule Learning from Knowledge Graphs

Mohamed H. Gad-Elrab[1(✉)], Daria Stepanova[1], Jacopo Urbani[2], and Gerhard Weikum[1]

[1] Max Planck Institute of Informatics, Saarbrücken, Germany
{gadelrab,dstepano,weikum}@mpi-inf.mpg.de
[2] VU University Amsterdam, Amsterdam, The Netherlands
jacopo@cs.vu.nl

Abstract. Advances in information extraction have enabled the automatic construction of large knowledge graphs (KGs) like DBpedia, Freebase, YAGO and Wikidata. These KGs are inevitably bound to be incomplete. To fill in the gaps, data correlations in the KG can be analyzed to infer Horn rules and to predict new facts. However, Horn rules do not take into account possible exceptions, so that predicting facts via such rules introduces errors. To overcome this problem, we present a method for effective revision of learned Horn rules by adding exceptions (i.e., negated atoms) into their bodies. This way errors are largely reduced. We apply our method to discover rules with exceptions from real-world KGs. Our experimental results demonstrate the effectiveness of the developed method and the improvements in accuracy for KG completion by rule-based fact prediction.

1 Introduction

Motivation and Problem. Recent advances in information extraction have led to huge graph-structured knowledge bases (KBs) also known as knowledge graphs (KGs) such as NELL [4], DBpedia [2], YAGO [22] and Wikidata [8]. These KGs contain millions or billions of relational facts in the form of subject-predicate-object (SPO) triples.

As such KGs are automatically constructed, they are incomplete and contain errors. To complete and curate a KG, inductive logic programming and data mining techniques (e.g., [5,11,29]) have been used to identify prominent patterns, such as *"Married people live in the same place"*, and cast them in the form of Horn rules, such as: $r_1 : livesIn(Y, Z) \leftarrow isMarriedTo(X, Y), livesIn(X, Z)$.

This has twofold benefits. First, since KGs operate under the Open World Assumption (OWA) (i.e., absent facts are treated as unknown rather than false), the rules can be used to derive additional facts. For example, applying the rule r_1 mined from the graph in Fig. 1a, the missing living place of Dave can be deduced based on the data about his wife Clara. Second, rules can be used to eliminate erroneous facts in the KG. For example, assuming that *livesIn* is a functional

P. Groth et al. (Eds.): ISWC 2016, Part I, LNCS 9981, pp. 234–251, 2016.
DOI: 10.1007/978-3-319-46523-4_15

relation, Amsterdam as a living place of Alice could be questioned as it differs from her husband's.

State of the Art and its Limitations. Methods for learning rules from KGs are typically based on inductive logic programming or association rule mining (see [11] and references given there). However, these methods are limited to Horn rules where all predicates in the rule body are positive. This is insufficient to capture rules that have exceptions, such as *"Married people live in the same place unless one is a researcher"*: r_2:*livesIn*$(Y, Z)\leftarrow$*isMarriedTo*(X, Y),*livesIn*(X, Z),*not researcher*(Y). This additional knowledge could be an explanation for Alice living in an unexpected place. If r_2 often holds, then one can no longer complete the missing living place for Dave by assuming that he lives with his wife Clara. Thus, understanding exceptions is crucial for KG completion and curation.

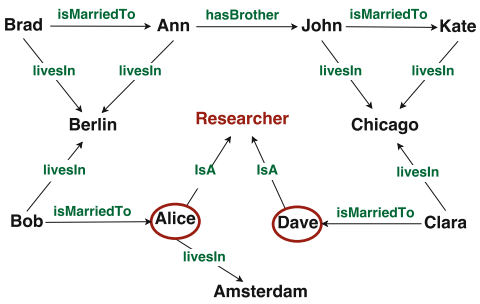
Our goal is to learn rules with exceptions, also known as *nonmonotonic rules*. Learning nonmonotonic rules under the Closed World Assumption (CWA) is a well-studied problem that lies at the intersection of inductive and abductive logic programming (e.g., [26,27]). However, these methods cannot be applied to KGs treated under the OWA.

Approach and Contribution. We present a novel method that takes a KG and a set of Horn rules as input and yields a set of exception-enriched rules as output. The output rules are no longer necessarily Horn clauses (e.g., rule r_2 above could be in our output). So we essentially we tackle a variant of a *theory revision* problem [30] under OWA.

Our method proceeds in four steps. First, we compute what we call "exception witnesses": predicates that are potentially involved in explaining exceptions (e.g., *researcher* in our example). Second, we generate nonmonotonic rule candidates that we could possibly add to our KG rules. Third, we devise quality measures for nonmonotonic rules to quantify their strength w.r.t the KG. In contrast to prior work, we do not merely give measures for individual rules in isolation, but also consider their cross-talk through a new technique that we call "partial materialization". Fourth and last, we rank the nonmonotonic rules by their strengths and choose a cut-off point such that the obtained rules describe the KG's content as well as possible with awareness of exceptions.

The salient contributions of our paper are:

- A framework for nonmonotonic rule mining as a knowledge revision task, to capture exceptions from Horn rules and overcome the limitations of prior work on KG rule mining.
- An algorithm for computing exception candidates, measuring their quality, and ranking them based on a novel technique that considers *partial materialization* of judiciously selected rules.
- Experiments with the YAGO3 and IMDB KGs where we show the gains of our method for rule quality as well as fact quality when performing KG completion.



(a) Rule mining for KG completion and KG cleaning

	$bornInUS$	$livesInUS$	$stateless$	$immigrant$	$singer$	$poet$	$hasUSPass$
$p1$	✓	✓			✓		✓
$p2$	✓	✓					✓
$p3$	✓	✓			✓	✓	✓
$p4$	✓	✓					✓
$p5$	✓	✓	✓				
$p6$	✓		✓				
$p7$	✓		✓				
$p8$	✓		✓	✓			
$p9$	✓			✓		✓	
$p10$	✓			✓	✓	✓	
$p11$	✓				✓	✓	✓

(b) US inhabitants KG

Fig. 1. Examples of knowledge graphs.

The rest of the paper is structured as follows. Section 2 introduces necessary notation and definitions. Section 3 presents our approach to nonmonotonic rule mining. Section 4 gives details on computing exceptions and revision candidates. Section 5 describes how we measure the quality of rules. Section 6 presents an experimental evaluation of how exception-enriched rules can improve the KG quality. Section 7 discusses related work.

2 Preliminaries

Knowledge Graphs. On the Web, knowledge graphs (KG) are often encoded using the RDF data model [16], which represents the content of the graph with a set of triples of the form $\langle subject\ predicate\ object \rangle$. These triples encode positive facts about the world, and they are naturally treated under the OWA.

In this work, we focus on KGs without blank nodes or schema (i.e. TBox). For simplicity, we represent the triples using unary and binary predicates. The unary predicates are the objects of the RDF *isA* predicate while the binary ones correspond to all other RDF predicates. We call this the factual representation $\mathcal{A}_\mathcal{G}$ (the subscript $\mathcal{G}$ is omitted when clear from context) of the KG $\mathcal{G}$ defined over the signature $\Sigma_{\mathcal{A}_\mathcal{G}} = \langle \mathbf{C}, \mathbf{R}, \mathcal{C} \rangle$, where $\mathbf{C}$, $\mathbf{R}$ and $\mathcal{C}$ are resp. sets of unary predicates, binary predicates and constants.

Example 1. The factual representation of the graph $\mathcal{G}$ from Fig. 1a among others contains the following facts:$ism(brad, ann)$;$ism(bob, alice)$;$li(brad, berlin)$; $r(alice)$;$r(dave)$; $hb(ann, john)$; $li(alice, amsterdam)$; $li(bob, berlin)$; $li(clara, chicago)$, where ism,li,hb,r stand for $isMarriedTo$, $livesIn$, $hasBrother$, and $researcher$ respectively. The signature of $\mathcal{A}_\mathcal{G}$ is $\Sigma_{\mathcal{A}_\mathcal{G}}=\langle \mathbf{C}, \mathbf{R}, \mathcal{C} \rangle$, where $\mathbf{C}=\{r\}$, $\mathbf{R}=\{ism, li, hb\}$ and $\mathcal{C}=\{john, ann, brad, dave, clara, alice, kate, bob, chicago, berlin, amsterdam\}$. □

In this work, we focus primarily on mining rules over unary predicates. Binary relations can be translated into multiple unary ones by concatenating the binary predicate and one of its arguments, e.g. the binary predicate *livesIn* of Fig. 1a can be translated into three unary ones *livesInAmsterdam, livesInBerlin, livesInChicago*. We apply this conversion to a KG in the input so that it consists of a collection of unary facts.

Nonmonotonic Logic Programs. Nonmonotonic Logic Programs We define a logic program in the usual way [21]. In short, a *(nonmonotonic) logic program* P is a set of *rules* of the form

$$H \leftarrow B, not \;\; E \tag{1}$$

where H is a standard first-order atom of the form $a(\boldsymbol{X})$ known as the rule head and denoted as $Head(r)$, B is a conjunction of positive atoms of the form $b_1(\boldsymbol{Y_1}), \ldots, b_k(\boldsymbol{Y_k})$ to which we refer as $Body^+(r)$ and $not \;\; E$, with slight abuse of notation, denotes the conjunction of atoms $not \;\; b_{k+1}(\boldsymbol{Y_{k+1}}), \ldots, not \;\; b_n(\boldsymbol{Y_n})$. Here, not is the so-called *negation as failure (NAF)* or *default negation*. The negated part of the body is denoted as $Body^-(r)$. The rule r is *positive* or *Horn* if $Body^-(r) = \emptyset$. $\boldsymbol{X}, \boldsymbol{Y_1}, \ldots, \boldsymbol{Y_n}$ are tuples of either constants or variables whose length corresponds to the arity of the predicates $a, b_1, \ldots, b_n$ respectively. The signature of P is given as $\Sigma_P = \langle \mathbf{P}, \mathcal{C} \rangle$, where $\mathbf{P}$ and $\mathcal{C}$ are resp. sets of predicates and constants occurring in P.

A logic program P is *ground* if it consists of only ground rules, i.e. rules without variables. Ground instantiation $Gr(P)$ of a nonground program P is obtained by substituting variables with constants in all possible ways. The *Herbrand universe* $HU(P)$ (resp. *Herbrand base* $HB(P)$) of P, is the set of all constants occurring in P, i.e. $HU(P) = \mathcal{C}$ (resp. the set of all possible ground atoms that can be formed with predicates in $\mathbf{P}$ and constants in $\mathcal{C}$). We refer to any subset of $HB(P)$ as a *Herbrand interpretation*. By $MM(P)$ we denote the set-inclusion minimal Herbrand interpretation of a ground positive program P.

An interpretation I of P is an *answer set* (or *stable model*) of P iff $I \in MM(P^I)$, where P^I is the *Gelfond-Lifschitz (GL) reduct* [12] of P, obtained from $Gr(P)$ by removing (i) each rule r such that $Body^-(r) \cap I \neq \emptyset$, and (ii) all the negative atoms from the remaining rules. The set of answer sets of a program P is denoted by $AS(P)$.

Example 2. Consider the program

$$P = \left\{ \begin{array}{l} (1)\ bornInUS(alex);\ (2)\ bornInUS(mat);\ (3)\ immigrant(mat); \\ (4)\ livesInUS(X) \leftarrow bornInUS(X), not \;\; immigrant(X) \end{array} \right\}$$

The ground instantiation $Gr(P)$ of P is obtained by substituting X with mat and $alex$. For I={$bornInUS(alex)$,$bornInUS(mat)$,$immigrant(mat)$, $livesInUS(alex)$}, the GL-reduct P^I of P contains the rule $livesInUS(alex) \leftarrow bornInUS(alex)$ and the facts (1)-(3). As I is a minimal model of P^I, it holds that I is an answer set of P. □

The answer set semantics for nonmonotonic logic programs is based on the CWA, under which whatever can not be derived from a program is assumed to be false. Nonmonotonic logic programs are widely applied for formalizing common sense reasoning from incomplete information.

Definition 1 (Rule-based KG completion). *Let $\mathcal{G}$ be a KG and $\mathcal{A}$ its factual representation over the signature $\Sigma_{\mathcal{A}} = \langle \mathbf{C}, \mathbf{R}, \mathcal{C} \rangle$. Let, moreover, $\mathcal{R}$ be a set of rules mined from $\mathcal{G}$, i.e. rules over the signature $\Sigma_{\mathcal{R}} = \langle \mathbf{C} \cup \mathbf{R}, \mathcal{C} \rangle$. Then* completion of $\mathcal{G}$ *(*resp. $\mathcal{A}$*) w.r.t. $\mathcal{R}$ is a graph $\mathcal{G}_{\mathcal{R}}$ constructed from any answer set $\mathcal{A}_{\mathcal{R}} \in AS(\mathcal{R} \cup \mathcal{A})$.*

Example 3. Consider a factual representation $\mathcal{A}$ of a KG $\mathcal{G}$ given in a tabular form in Fig. 1b, where a tick appears in an intersection of a row s and a column o, if $o(s) \in \mathcal{A}$ (resp. $\langle s \quad isA \quad o \rangle \in \mathcal{G}$). Suppose we are given a set of rules $\mathcal{R} = \{r_1, r_2\}$, where

$r_1 : livesInUS(X) \leftarrow bornInUS(X), not \quad immigrant(X);$
$r_2 : livesInUS(X) \leftarrow hasUSPass(X).$

The program $\mathcal{A} \cup \mathcal{R}$ has a single answer set $\mathcal{A}_{\mathcal{R}}$=$\mathcal{A} \cup \{livesInUS(p_i) \mid i$=$6, 7, 11\}$, from which the completion $\mathcal{G}_{\mathcal{R}}$ of $\mathcal{G}$ can be reconstructed. □

3 Learning Exception-Enriched Rules

Horn Rule Revision. Before we formally define our problem, we introduce the notion of an incomplete data source following [7].

Definition 2 (Incomplete data source). *An incomplete data source is a pair $G = (\mathcal{G}^a, \mathcal{G}^i)$ of two KGs, where $\mathcal{G}^a \subseteq \mathcal{G}^i$ and $\Sigma_{\mathcal{A}_{\mathcal{G}^a}} = \Sigma_{\mathcal{A}_{\mathcal{G}^i}}$. We call $\mathcal{G}^a$ the available graph and $\mathcal{G}^i$ the ideal graph.*

The graph $\mathcal{G}^a$ is the graph that we have available as input. The ideal graph $\mathcal{G}^i$ is the perfect completion of $\mathcal{G}^a$, which is supposed to contain all correct facts with entities and relations from $\Sigma_{\mathcal{A}_{\mathcal{G}^a}}$ that hold in the current state of the world.

Given a potentially incomplete graph $\mathcal{G}^a$ and a set of Horn rules $\mathcal{R}_H$ mined from $\mathcal{G}^a$, our goal is to add default negated atoms (exceptions) to the rules in $\mathcal{R}_H$ and obtain a revised ruleset $\mathcal{R}_{NM}$ such that the set difference between $\mathcal{G}^a_{\mathcal{R}_{NM}}$ and $\mathcal{G}^i$ is smaller than between $\mathcal{G}^a_{\mathcal{R}_H}$ and $\mathcal{G}^i$. If in addition the set difference between $\mathcal{G}^a_{\mathcal{R}_{NM}}$ and $\mathcal{G}^i$ is the smallest among the ones produced by other revisions $\mathcal{R}'_{NM}$ of $\mathcal{R}_H$, then we call $\mathcal{R}_{NM}$ an *ideal nonmonotonic* revision. For single rules such revision is defined as follows:

Definition 3 (Ideal nonmonotonic revision). *Let $G = (\mathcal{G}^a, \mathcal{G}^i)$ be an incomplete data source. Moreover, let $r : \; a \leftarrow b_1, \ldots, b_k$ be a Horn rule mined from $\mathcal{G}^a$. An* ideal nonmonotonic revision *of r w.r.t. $\mathcal{G}$ is any rule*

$$r' : \; a \leftarrow b_1, \ldots, b_k, not \quad b_{k+1}, not \quad b_n, \tag{2}$$

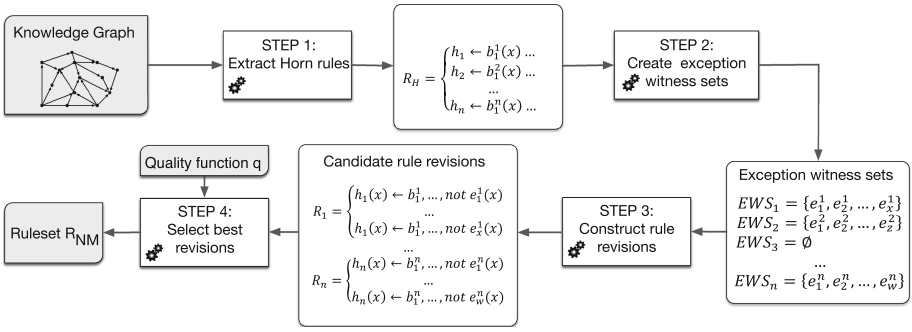


Fig. 2. Exception-enriched rule learning: general overview

such that $\mathcal{G}^i \triangle \mathcal{G}^a_{r'} \subset \mathcal{G}^i \triangle \mathcal{G}^a_r$ [1], *i.e. the completion of* $\mathcal{G}^a$ *based on* r' *is closer to* $\mathcal{G}^i$ *than the completion of* $\mathcal{G}^a$ *based on* r, *and* $\mathcal{G}^a_{r''} \triangle \mathcal{G}^i \subset \mathcal{G}^a_{r'} \triangle \mathcal{G}^i$ *for no other nonmonotonic revision* $r'' \neq r'$ *of* r. *If k=n, then the revision coincides with the original rule.*

In our work, we assume that the ideal graph $\mathcal{G}^i$ is not available (otherwise nothing would need to be learnt). Therefore, we cannot verify whether a revision is ideal for $\mathcal{R}_H$. What we can do, however, is to estimate using some quality functions whether a given revision produces an approximation of $\mathcal{G}^i$ that is better than the approximation produced by the original Horn ruleset. For this purpose, we introduce a generic quality function q which receives as input a revision $\mathcal{R}_{NM}$ of the ruleset $\mathcal{R}_H$ and a graph $\mathcal{G}$, and returns a real value that reflects the quality of the revised set $\mathcal{R}_{NM}$. We can now formally define our problem:

Problem: quality-based Horn rule revision

Given: KG $\mathcal{G}$, set of nonground Horn rules $\mathcal{R}_H$ mined from $\mathcal{G}$, quality function q

Find: set of rules $\mathcal{R}_{NM}$ obtained by adding default negated atoms to $Body^-(r)$ for some $r \in \mathcal{R}_H$, such that $q(\mathcal{R}_{NM}, \mathcal{G})$ is maximal.

Note that so far we did not specify the details of the quality function q. In our approach, we estimate the quality of a ruleset by exploiting well-established measures proposed in the field of data mining [3]. Even though none of these measures can offer any sort of guarantee, our hypothesis is that they still indicate to some extent the percentage of correctly predicted facts obtained as a result of completing a KG based on a given ruleset. We discuss in Sect. 5 in more details how q can be defined.

Approach Overview. Figure 2 illustrates the main phases of our approach. In Step 1, we launch an off-the-shelf algorithm to mine Horn rules from the input KG. We use FPGrowth [13], but any other, e.g., [5,11] can be likewise applied, i.e., our overall revision approach is independent of the concrete technique used

[1] $\mathcal{G}_1 \triangle \mathcal{G}_2 = (\mathcal{G}_1 \backslash \mathcal{G}_2) \cup (\mathcal{G}_2 \backslash \mathcal{G}_1)$.

Algorithm 1: *ComputeEWS*: compute $EWS(r, \mathcal{A})$

Input: KB $\mathcal{A}$, rule $r: a(X) \leftarrow b_1(X), \ldots, b_k(X)$
Output: $EWS(r, \mathcal{A})$
(a) $N \leftarrow NS(r, \mathcal{A})$; $A \leftarrow ABS(r, \mathcal{A})$
(b) $E^+ \leftarrow \{not\_a(c) \mid c \in A\}$; $E^- \leftarrow \{not\_a(c) \mid c \in N\}$
(c) $\mathcal{R}_e \leftarrow Learn(E^+, E^-, \mathcal{A})$
(d) $EWS \leftarrow \{\text{predicate } p \text{ in } Body^+(r') \mid r' \in \mathcal{R}_e, \text{ s.t. }, p \text{ is not in } Body^+(r)\}$
(e) **return** EWS

for Horn rule mining. Then, for each rule we compute *normal* and *abnormal* instance sets, defined as:

Definition 4 (r-(ab)normal instance set). *Let $\mathcal{A}$ be the factual representation of a KG $\mathcal{G}$ and $r: a(X) \leftarrow b_1(X), \ldots, b_k(X)$ a Horn rule mined from $\mathcal{G}$. Then,*

- $NS(r, \mathcal{A}) = \{c \mid b_1(c), \ldots, b_k(c), a(c) \in \mathcal{A}\}$ *is an* r-normal instance set*;*
- $ABS(r, \mathcal{A}) = \{c \mid b_1(c), \ldots, b_k(c) \in \mathcal{A}, a(c) \notin \mathcal{A}\}$ *is an* r-abnormal instance set*.*

Example 4. For $\mathcal{A}$ from Fig. 1b and the rule $r: livesInUS(X) \leftarrow bornInUS(X)$, r-normal and r-abnormal instance sets are given as $NS(r, \mathcal{A}) = \{p1, \ldots, p5\}$ and $ABS(r, \mathcal{A}) = \{p6, \ldots, p11\}$ respectively. □

Intuitively, if the given data was complete, then the r-normal and r-abnormal instance sets would exactly correspond to instances for which the rule r holds (resp. does not hold) in the real world. Since the KG is potentially incomplete, this is no longer the case and some r-abnormal instances might in fact be classified as such due to data incompleteness. In order to distinguish between the "wrongly" and "correctly" classified instances in the r-abnormal set, in Step 2 we construct *exception witness sets* (EWS), which are defined as follows:

Definition 5 (Exception witness set (EWS)). *Let $\mathcal{A}$ be the factual representation of a KG $\mathcal{G}$ and let r be a Horn rule mined from $\mathcal{G}$. An* r-exception witness set $EWS(r, \mathcal{A}) = \{e_1, \ldots, e_l\}$ *is a maximal set of predicates, such that*

(i) $e_i(c') \in \mathcal{A}$ *for some* $c' \in ABS(r, \mathcal{A})$, $1 \leq i \leq m$ *and*
(ii) $e_1(c), \ldots, e_m(c) \notin \mathcal{A}$ *for all* $c \in NS(r, \mathcal{A})$.

Example 5. For $\mathcal{A}$ and r from Example 4 $EWS(r, \mathcal{A}) = \{immigrant\}$ is an r-exception witness set. For $\mathcal{A}' = \mathcal{A} \setminus \{p5\}$ it holds that $EWS(r, \mathcal{A}') = \{immigrant, stateless\}$. □

After EWSs are computed for all rules in $\mathcal{R}_H$, we use them to create potential revisions in Step 3. Then, we rank the newly created revisions and select the best ones using different criteria (Step 4). These selected rules will constitute the new $\mathcal{R}_{NM}$.

4 Computing Exception Witnesses and Potential Rule Revisions

In this section we describe how we calculate the exception witness sets for Horn rules (Fig. 2, Step 2) and how we create potential rule revisions (Fig. 2, Step 3).

Computing Exception Witness Sets. For constructing exception witness sets we use the algorithm *ComputeEWS* (Algorithm 1), which given a factual representation $\mathcal{A}$ of a KG and a rule $r \in \mathcal{R}_H$ as input, outputs the set $EWS(r, \mathcal{A})$.

The algorithm works as follows: First in (a) r-normal $NS(r, \mathcal{A})$ and r-abnormal $ABS(r, \mathcal{A})$ instance sets are found and stored resp. in N and A. Then in (b) the fresh predicate not_a the facts $not\_a(c)$ are added to E^+ for all $c \in ABS(r, \mathcal{A})$. In the same step the facts $not\_a(c)$ for $c \in N$ are stored in E^-. In (c), a variant of a classical inductive learning procedure $Learn(E^+, E^-, \mathcal{A})$, e.g., [23] is employed to induce a set of hypothesis $\mathcal{R}_e$ in the form of Horn rules with unary atoms, s.t. $\mathcal{A} \cup \mathcal{R}_e \models e$ for as many as possible $e \in E^+$, and $\mathcal{A} \cup \mathcal{R}_e \not\models e'$ for all $e' \in E^-$. Finally, in (d) the bodies of rules in $\mathcal{R}_e$ not containing predicates from $Body^+(r)$ are put in EWS, which is output in (e).

The correctness of *ComputeEWS* follows from the correctness of the procedure *Learn*. Indeed, by (d) for $p \in EWS$, a rule r' with p occurring in $Body(r')$ exists in $\mathcal{R}_e$. Since $r' \cup \mathcal{A} \not\models not\_a(c)$ for $not\_a(c) \in E^-$, we have that $p(c) \notin \mathcal{A}$ for r-normal c due to (a) and (b). Moreover, $p(c') \in \mathcal{A}$ for some r-abnormal c', as otherwise $r' \notin \mathcal{R}_e$. Hence, (i) and (ii) of Definition 5 hold, i.e. EWS is an exception witness set for r w.r.t. $\mathcal{A}$.

Constructing Candidate Rule Revisions. After all EWSs are calculated for Horn rules in $\mathcal{R}_H$, we construct a search space of potential revisions by adding to rule bodies exceptions in the form of default negated atoms. More specifically, for every $r_i : a(X) \leftarrow b_1(X), \ldots, b_k(X)$ in $\mathcal{R}_H$ we create $m = |EWS(r_i, \mathcal{A})|$ revision candidates, i.e. rules $r_i^{e_j}$, s.t. $Head(r_i^{e_j}) = Head(r_i)$, $Body^+(r_i^{e_j}) = Body(r_i)$, $Body^-(r_i) = e_j(X)$, where $e_j \in EWS(r_i, \mathcal{A})$. We denote with $\mathcal{R}_i$ the set of all $r_i^{e_j}$.

Example 6. For $EWS(r, \mathcal{A}') = \{immigrant, stateless\}$ from Example 5 in Step 3 revision candidates $r^{im} : livesInUS(X) \leftarrow bornInUS(X), not\ \ immigrant(X)$ and $r^{st} : livesInUS(X) \leftarrow bornInUS(X), not\ \ stateless(X)$ are created. □

5 Rules Quality Assessment

Given a potential R_{NM}, the function q should approximate the closeness between the completion $\mathcal{G}^a_{\mathcal{R}_{NM}}$ of the input KG $\mathcal{G}^a$ and the ideal KG $\mathcal{G}^i$. In this work, we follow usual practice in data mining and adapt standard association rule measures to our needs. Let rm be a generic rule measure, e.g. one defined in Table 1[2]. Then, naively generalizing rm for rulesets by taking the average of rm

[2] Table 1 reports the definition of confidence, lift and Jaccard coefficient – three commonly-used rule measures [1]. Here, $n(B)$ (resp. $n(H)$) denotes the number of

Table 1. Rule evaluation measures for a rule r w.r.t. $\mathcal{A}$

Rule measure	Formula for $r: H \leftarrow B$
Confidence	$conf(r, \mathcal{A}) = \frac{n(HB)}{n(B)}$
Lift	$lift(r, \mathcal{A}) = \frac{n(HB)}{n(H) * n(B)}$
Jaccard coef	$jc(r, \mathcal{A}) = \frac{n(HB)}{n(H) + n(B) - n(HB)}$

values for all rules in a given set we obtain

$$q_{rm}(\mathcal{R}_{NM}, \mathcal{A}) = \frac{\sum_{r \in \mathcal{R}_{NM}} rm(r, \mathcal{A})}{|\mathcal{R}_{NM}|} \tag{3}$$

In our case, q_{rm} alone is not sufficiently representative for being the target quality function q for two reasons: (1) it does not penalize rules with noisy exceptions[3]; (2) it does not measure how many contradicting beliefs our revisions reflect.

Example 7.

(1) For $r: livesInUS(X) \leftarrow hasUSPass(X), not\ \ poet(X)$ and $\mathcal{A}$ (from Fig. 1b) we have $conf(r, \mathcal{A})$=1, as all 3 non-poets with US passports live in the US, i.e., r gets the highest individual score based on confidence. However, *poet* is a noisy exception due to p_3, who is a poet possessing a US passport and living in the US.

(2) Let $\mathcal{R}_{NM} = \{r_1 : lu(X) \leftarrow hu(X), st(X), r_2 : lu(X) \leftarrow bu(X), not\ \ im(X), r_3 : im(X) \leftarrow st(X)\}$, where lu, hu, bu, st, im stand for $livesInUS, hasUSPass$, $bornInUS$, $stateless$ and $immigrant$. Although im in r_2 may perfectly fit as exception w.r.t. some (unspecified here) original KG; once the KG is completed based on r_1 and r_3, im might become noisy for r_2. Indeed, r_1 can easily bring new instances c in lu, while r_3 can predict facts $im(c)$. If this is the case, i.e., $r_2 \in \mathcal{R}_{NM}$ becomes noisy after other rules in $\mathcal{R}_{NM}$ are applied, then intuitively rules in $\mathcal{R}_{NM}$ do not agree on the beliefs about $\mathcal{G}^i$ they express.

□

To resolve the above issues we introduce an additional quality function $q_{conflict}$, next to q_{rm}, whose purpose is to evaluate the ruleset w.r.t (1) and (2). To measure $q_{conflict}$ for $\mathcal{R}_{NM}$, we create an extended set of rules $\mathcal{R}^{aux}$, which contains every revised rule $r: a(X) \leftarrow b(X), not\ \ e(X)$ in $\mathcal{R}_{NM}$ and its auxiliary version $r_{aux} : not\_a(X) \leftarrow b(X), e(X)$, where not_a is a fresh predicate collecting instances that are not in a. Notice that r_{aux} is meaningless, and thus

instances for which the body (resp. head) of a rule $H \leftarrow B$ is satisfied in $\mathcal{A}$ or in data mining terminology the number of transactions in $\mathcal{A}$ with items from B (resp. H).

[3] e is a *noisy* exception for r if $e(c) \in \mathcal{A}$ for some r-normal c.

void in $\mathcal{R}^{aux}$, for rules r with positive bodies. Formally, we define $q_{conflict}$ as follows

$$q_{conflict}(\mathcal{R}_{NM}, \mathcal{A}) = \sum_{p \in pred(\mathcal{R}^{aux})} \frac{|\{c \,|\, p(c), not\_p(c) \in \mathcal{A}_{\mathcal{R}^{aux}}\}|}{|\{c \,|\, not\_p(c) \in \mathcal{A}_{\mathcal{R}^{aux}}\}|} \tag{4}$$

where $pred(\mathcal{R}^{aux})$ is the set of predicates appearing in $\mathcal{R}^{aux}$.

Intuitively, $\mathcal{A}_{\mathcal{R}^{aux}}$ contains both positive predictions of the form $p(c)$ and negative ones $not\_p(c)$ produced by the rules in $\mathcal{R}^{aux}$. The function $q_{conflict}$ computes the ratio of "contradicting" pairs $\{p(c), not\_p(c)\}$ over the number of $not\_p(c)$[4] in $\mathcal{A}_{\mathcal{R}^{aux}}$, which reflects how much the rules in $\mathcal{R}_{NM}$ disagree with each other on beliefs about the ideal KG $\mathcal{G}^i$ they express. The smaller $q_{conflict}$, the better is the ruleset $\mathcal{R}_{NM}$.

Revision Based on Partial Materialization. Our goal in Step 4 is to find a set of revisions $\mathcal{R}_{NM}$, for which $q_{rm}(\mathcal{R}_{NM}, \mathcal{A})$ is maximal and $q_{conflict}(\mathcal{R}_{NM}, \mathcal{A})$ is minimal.

To determine such globally best set $\mathcal{R}_{NM}$ many candidate rule combinations have to be checked, which is unfortunately not feasible because of the large size of our $\mathcal{A}$ and EWS. Therefore, we propose an approach where we incrementally build $\mathcal{R}_{NM}$ by considering every $r_i \in \mathcal{R}_H$ and choose the best revision $r_i^j \in \mathcal{R}_i$ for it. In order to select the best r_i^j, we use a special ranking function, which estimates how well a rule r at hand describes the data and how noisy its exceptions are. In the remaining of this section, we will propose four different ranking functions, starting from the simplest to the most sophisticated one.

Naive-**ranker**. The first implementation, which we call *rank_naive*, calculates the average value of the rm scores of r and r_{aux} and uses it to rank the rules. Formally, the average is computed by the following function:

$$est_{rm}(r, \mathcal{A}) = \frac{rm(\overbrace{H \leftarrow B, not\ E}^{r}, \mathcal{A}) + rm(\overbrace{not\_H \leftarrow B, E}^{r_{aux}}, \mathcal{A})}{2} \tag{5}$$

where rm is one of the measures in Table 1. E.g., plugging in *conf* instead of rm, gives

$$est_{conf}(r, \mathcal{A}) = \frac{1}{2}\Big(\frac{n(BH) - n(BHE)}{n(B) - n(BE)} + \frac{n(BE) - n(BHE)}{n(BE)}\Big) \tag{6}$$

where $n(X)$ is the number of transactions with items from X.

Example 8. For r and $\mathcal{A}$ from Example 7 (1) $est_{conf}(r, \mathcal{A}) = 0.75$, i.e., due to noisiness of *poet* the value of est_{conf} decreased. □

PM-**ranker**. The main problem of *rank_naive* is that it does not exploit any knowledge about the properties that a final revision $\mathcal{R}_{NM}$ might have. In other

[4] Ratio over the number of $p(c)$ instead of $not\_p(c)$ is possible, but then $q_{conflict}$ is smaller and less representative.

words, ranking of revisions of a rule at hand is completely independent from ranking of revisions for other rules. To address this issue, we propose a second implementation called *revision based on partial materialization* (denoted as *rank_pm*). Here, the idea is to apply est_{rm} for a rule r not on $\mathcal{A}$ but on completion of $\mathcal{A}$ based on other rules, which according to our estimates constitute some approximation of $\mathcal{R}_{NM}$.

Example 9. Consider a rule r_1:$lu(X)\leftarrow bu(X), not \quad im(X)$, and suppose there is only a single other rule r_2:$lu(X)\leftarrow hu(X)$ given, for which $EWS(r_2, \mathcal{A}) = \emptyset$ for $\mathcal{A}$ from Fig. 1b. This knowledge can be exploited when ranking r_1. We have $est_{conf}(r_1, \mathcal{A}) = 0.8$, while $est_{conf}(r_1, \mathcal{A}_{r_2}) = 0.875$ due to the materialized fact $livesInUS(p11)$. This increase gives us an indication that r_1 agrees with r_2 on predictions it makes.

On the contrary, for $r_3 : lu(X) \leftarrow hu(X), not \quad pt(X)\}$, where pt stands for *poet* and $r_4 : lu(X)\leftarrow bu(X)$ we have $est_{conf}(r_3, \mathcal{A})$=0.75, but $est_{conf}(r_3, \mathcal{A}_{r_4})$=0.5, which witnesses that beliefs of r_3 and r_4 contradict. □

The function *rank_pm* first constructs the temporary rule set $\mathcal{R}^t$, which contains, for every rule $r_i \in \mathcal{R}_H$, a rule r_i^t with all exceptions from $EWS(r_i, \mathcal{A})$ incorporated, i.e., $\mathcal{R}^t$ predicts the smallest number of facts, which are also predicted by any possible revision $\mathcal{R}_{NM}$. Then, for each $r_i \in \mathcal{R}_H$, we compute the est_{rm} value for all revision candidates r_i^j based on $\mathcal{A}_{\mathcal{R}^t \setminus r_i^t}$. Formally,

$$rank\_pm(r_i^j, \mathcal{A}) = est_{rm}(r_i^j, \mathcal{A}_{\mathcal{R}^t \setminus r_i^t}) \tag{7}$$

Once the scores for all revision candidates r_i^j for r_i are computed, we pick the revision with the highest score, add it to the current snapshot of $\mathcal{R}_{NM}$ and move to r_{i+1}.

OPM-**ranker**. With *rank_pm*, facts inferred by rules of low quality might have a significant impact on more promising rules. To handle this issue, we propose a variation of *rank_pm* called *revision with ordered partial materialization* (abbr. *rank_opm*), which proceeds as follows. First we rank Horn rules based on some rm' (possibly same as rm) and obtain an ordered list $os_{\mathcal{R}_H}$. Then we go through $os_{\mathcal{R}_H}$ and for every rule r_i we compute a snapshot $\mathcal{A}_i$ of $\mathcal{A}$ by materializing only those rules $r_k^t \in \mathcal{R}^t$, for which r_k is ordered higher in the list $os_{\mathcal{R}_H}$ than r_i. More formally,

$$rank\_opm_{rm}(r_i, \mathcal{A}) = est_{rm}(r_i, \mathcal{A}_i) \tag{8}$$

where $\mathcal{A}_i = \mathcal{A}_{\mathcal{R}^t \setminus \{r_k^t \mid os_{\mathcal{R}_H}[k]=r_k;\, i \geq k\}}$.

OWPM-**ranker**. With *rank_opm* as we have defined it, the facts inferred by rules count the same as the true facts in $\mathcal{A}$. Since the predicted facts are inferred based on statistically-supported assumptions, it is natural to distinguish them from the facts that are explicitly present in $\mathcal{A}$. To achieve this, we propose one last ranking function that exploits weights assigned to facts. Here, there is a clear distinction between facts from $\mathcal{A}$ (which get maximal weight) and the predicted facts (which inherit weights from rules that inferred them). We call this method *revision with ordered weighted partial materialization* (abbr. *rank_owpm*).

The method *rank_owpm* differs from *rank_opm* in that weights are used to estimate the revisions' scores. It is convenient (and a common practice) to assign weights of probabilistic nature between 0 and 1 (e.g., *confidence* can be exploited). There are several ways to produce weighted partial materialization; for example, using probabilistic logic programming systems, e.g., Problog [9] or PrASP [25].

However, normally, in such systems facts predicted by some rules in a ruleset at hand are used as input to other rules, i.e., uncertainty is propagated through rule chains, which might be undesired in our setting. To avoid such propagation, when computing weighted partial materialization of $\mathcal{A}$ we keep predicted facts (i.e., derived using rules) separately from the explicit facts (i.e., those in $\mathcal{A}$), and infer new facts using only $\mathcal{A}$.

The method *rank_owpm* works as follows. Initially, we sort the rules in $\mathcal{R}_H$ and create the $\mathcal{A}_i$s with the same procedure as described for *rank_opm*. The only difference is that here every inferred fact in $\mathcal{A}_{\mathcal{R}^t}$ receives a specific weight that corresponds to $rm(r', \mathcal{A})$, where r' is the positive version of the rule that inferred the fact[5]. If the same fact is derived by multiple rules, we keep the highest weight.

The weights play a role when we evaluate a rule w.r.t. the partially materialized KG. To this end, we slightly change the *rm* function so that it considers weighted facts (we denote such function as rm^w). E.g., $conf^w(r, \mathcal{A})$ calculates a weighted sum of the instances for which the head (resp. body) of r is satisfied w.r.t. $\mathcal{A}$ (instead of a normal sum used in *conf*). Formally, *rank_owpm* computes a score for a revision r_i^j as follows:

$$rank\_owpm_{rm^w}(r_i^j, \mathcal{A}) = est_{rm^w}(r_i^j, \mathcal{A}_i^w) \tag{9}$$

where $\mathcal{A}_i^w$ is the weighted version of $\mathcal{A}_i$ from Eq. 8. In the following section, we will analyze the performance of these four functions on some realistic KGs.

6 Evaluation

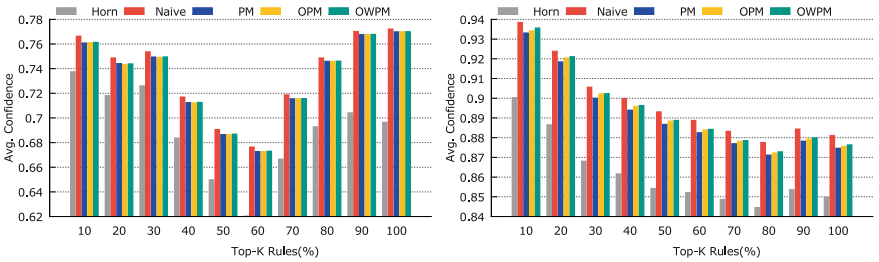
Experimental Setup. We considered two knowledge graphs: a slice of almost 10M facts from YAGO3 [22], a general purpose KG, and an RDF version of IMDB[6] data with 2M facts, a well known domain-specific KG of movies and artists. We chose these two KGs in order to evaluate our method's performance on both general-purpose and domain-specific KGs. Our experiments were performed on a machine with 40 cores and 400GB RAM. The used datasets and the experimental code are publicly available[7].

Outline. First we evaluate different configurations of our method using the quality functions q_{rm} and $q_{conflict}$, defined in Sect. 5. Then, we report the results of a

[5] We cannot consider the entire rule (i.e. with all exceptions attached), since standard measures like $confidence$ will return values very close to 1 for such rules.

[6] http://imdb.com.

[7] http://people.mpi-inf.mpg.de/~gadelrab/rules_iswc.



(a) Confidence for top-k YAGO revised rules (b) Confidence for top-k IMDB revised rules

Fig. 3. Average rules' confidence on YAGO and IMDB (higher is better).

manual assessment that we performed to evaluate the quality of the predictions, reporting good and bad examples produced by our method.

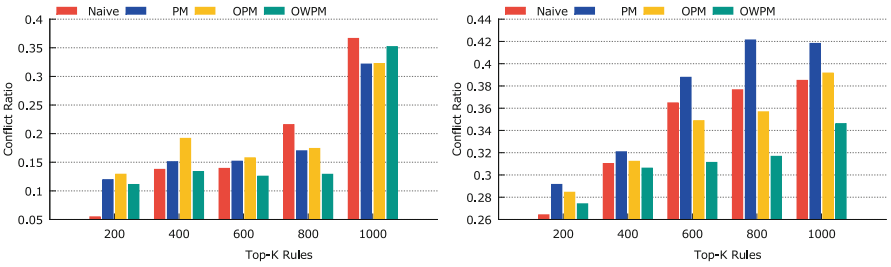
6.1 Evaluation of the Revision Steps

Step 1. Initially, we considered the Horn rules produced by AMIE [11]. However, they mainly focus on unsupported binary predicates and the only unary rules are restricted to the *isA* predicate, which was too limiting for us. Therefore, we first propositionalized the original KG, and then mined the Horn rules using the association rule mining implementation based on standard FPGrowth [13] offered by SPMF Library[8]. In order to avoid over-fitting rules as well as to reduce the computation, we limited the extraction to rules with maximum four body atoms, a single head atom, a minimum support of $0.0001 \times \#$ *entities* and a minimum confidence of 0.25 for YAGO. Since IMDB is smaller and more connected, we set a higher minimum support of $0.005 \times \#$ *entities* and confidence of 0.6. On our machine, this process took approx. 10 seconds on YAGO and 2.5 second on IMDB, and it generated about 10 K and 25K rules respectively.

Steps 2 and 3. We implemented a simple inductive learning procedure, which performs manipulations on the set of facts instantiating the rule and its body to get the EWS. The generation of EWSs with minimum support of 0.05 took about 50 seconds for YAGO and 30 seconds for IMDB. The execution time is significantly affected by the size and distribution of the predicates in the KG. We could find EWSs for about 6 K rules mined from YAGO, and 22 K rules mined from IMDB. On average, the EWSs for the YAGO's rules contained 3 exceptions, and 28 exceptions on IMDB.

Step 4. We evaluated the quality of our rule selection procedure w.r.t. two dimensions, which reflect the two q proposed in Sect. 5: *average of the rules' confidence* (q_{conf}), and the *number of conflicts* ($q_{conflict}$). The average confidence shows how well the revised rules adhere to the input. The number of conflicts indicates how consistent the revised rules set is w.r.t the final predictions

[8] http://www.philippe-fournier-viger.com/spmf/.



(a) Ratio of conflicts for YAGO rules

(b) Ratio of conflicts for IMDB rules

Fig. 4. Ratio of conflicts on YAGO and IMDB (lower is better).

it makes. Due to space constraints, we report the results using only confidence as rule evaluation function (Eq. 6) and lift as rule ordering criterion, as we found this combination to be a good representative.

6.2 Exception-Enriched Rules vs. Horn Rules

Figure 3 reports the obtained average rules' confidence using the four ranking functions to select the best revisions. *Horn* reports the average confidence of the original Horn rules; while *Naive*, *PM*, *OPM* and *OWPM* are our ranking methods described in Sect. 5. For both inputs, we show the results on the top $10, \ldots, 100\%$ rules ranked by lift.

We make three observations. (i) In general enriching Horn rules with exceptions increases the average confidence (approx. 11 % for YAGO, 3.5 % for IMDB). This indicates that our method is useful to mine rules that reflect the data more precisely. It is also worth mentioning that along with the increase in confidence, the average coverage of the revised rules dropped only by 13 % for YAGO and 4 % for IMDB (i.e. the rules do not become too specific). (ii) The comparison between the four ranking methods shows that the highest confidence is achieved by the non-materialized (*Naive*) function followed by the weighted one (*OWPM*). (iii) Since we used lift for ordering the rules, and it is not neccessarily correlated with confidence, one can see that the confidence drops for around top 60 % of the YAGO rules, and then slightly increases again. For IMDB a smooth confidence decrease is observed with the addition of lower-ranked rules.

The higher value of *Naive* was expected, since this procedure is designed to maximize the confidence. However, confidence alone is not a good indicator to determine the overall rule's quality, as we explained in Sect. 5. Figure 4 shows the number of conflicts (for YAGO and IMDB) that were obtained by executing the revised rules and their corresponding auxiliary versions (r_{aux}) using the DLV system [19]. Unfortunately, DLV was unable to scale to the entire ruleset; hence, we used up to 1000 rules. In our experiment, a conflict occurs when we derive both $p(c)$ and $not\_p(c)$. The graphs report the ratio between the number of conflicts and negated derived facts. From them, we observe that both *OPM* and

$Y_1 : isMountain(X) \leftarrow isLocatedInAustria(X), isLocatedInItaly(X),$
$\quad \textbf{not}[isRiver(X)|isLocatedInRussia(X)]$
$Y_2 : bornInUSA(X) \leftarrow actedInMovie(X), createdMovie(X), isPerson(X),$
$\quad \textbf{not}[wonFilmfareAwards(X)|bornInNewYork(X)]$
$Y_3 : isPoliticianOfUSA(X) \leftarrow bornInUSA(X), isGovernor(X),$
$\quad \textbf{not}[isPoliticianOfPuertoRico(X)|isPoliticianOfHawaii(X)]$

$I_1 : hasLanguageEnglish(X) \leftarrow hasGenreDrama(X), hasGenreTriller(X), hasGenreCrime(X),$
$\quad \textbf{not}[producedInIndia(X)|createdByNovelist(X)]$
$I_2 : hasGenreAnimation(X) \leftarrow directedByActor(X), hasLanguageEnglish(X), producedInUSA(X),$
$\quad hasGenreFamily(X), \textbf{not}[hasGenreDrama(X)|producedIn1984(X)]$

Fig. 5. Anecdotal example rules (Y = YAGO, I = IMDB) with good and bad exceptions

OWPM produce less conflicts than the *Naive* function in most of the cases. By comparing the *OPM* and *OWPM* functions, we find that the weighted version is better, especially on the IMDB dataset when we can reduce the conflicts from 775 to 685 on a base of about 2000 negated facts.

We executed the top-1000 revised rules using DLV and counted the number of derivations that our exceptions prevented. For YAGO with the original Horn rules, the reasoner inferred 924591 new triples. Our exception-enriched ruleset decreased the number of inferred triples to 888215 (*Naive*), 892707 (*PM*), 892399 (*OPM*), and 891007 (*OWPM*). For IMDB we observed a smaller reduction. With the Horn rules the reasoner derived 38609 triples, while with the revised rules the inference set decreased to 36069 (*Naive*), 36355 (*PM*), 36021 (*OPM*), and 36028 (*OWPM*) triples.

Unfortunately, there is no automatic way available to assess whether the removed inference consists of genuine errors. Therefore, we selected the revised ruleset produced by the *OWPM* function and sampled 259 random facts from YAGO (we selected three facts for each binary predicate to avoid skewness). Then, we manually consulted online resources like Wikipedia to determine whether these triples were indeed incorrect. We found that 74.3 % of these triples consisted of factual mistakes. This number provides a first empirical evidence that our method is indeed capable of detecting good exceptions and hence can improve the general quality of the Horn rules.

We conclude reporting some anecdotal examples of rules on YAGO and IMDB in Fig. 5. Between the brackets we show examples of both good (underlined) and bad exceptions. In some cases, the rules have high quality exceptions such as rule $Y1$. In others, we found that the highest ranked exceptions mainly refer to disjoint classes of the head. The complete list of mined rules with the scores given to the determined exceptions is available in our repository.

7 Related Work

The problem of automatically learning patterns from KGs and exploiting them for predicting new facts has gained a lot of attention in the recent years. Approaches for predicting unseen data in KGs can be roughly divided into

two groups: statistics-based, and logic-based. The firsts apply well-known techniques like tensor factorization, or neural-embedding-based models (see [24] for overview). The second group focuses more on logical rule learning (e.g., [11,29]). The most relevant works for us are in the last group. These, however, typically focus on learning Horn rules, rather than nonmonotonic (i.e., exception-enriched) as we do.

In the association rule mining community, some works concenrated on finding (interesting) exception rules (e.g. [28]), which are defined as rules with low support (rare) and high confidence. Our work differs from this line of research because we do not necessarily look for rare interesting rules, but care about the quality of their predictions. Another relevant stream of research is concerned with learning Description Logic TBoxes or schema (e.g., [18]). However, these techniques focus on learning concept definitions rather than nonmonotonic rules.

In the context of inductive and abductive logic, learning nonmonotonic rules from complete datasets [10] was studied in several works ([6,15,17,26,26,27]. These methods rely on CWA and focus on describing a dataset at hand exploiting negative example, which are explicitly given unlike in our setting.

Learning Horn rules in presence of incompleteness was studied in hybrid settings in [14,20]. There a background theory or a hypothesis can be represented as a combination of a DL ontology and Horn rules. While the focus of this work is on the complex interaction between reasoning components and the learned rules are positive, we are concerned with techniques for deriving nonmonotonic rules with high predictive quality from huge KGs.

8 Conclusions and Future Work

We have presented a method for mining nonmonotonic rules from KGs: first learning a set of Horn rules, and then revising them by adding negated atoms into their bodies with the goal of improving the quality of a rule set for data prediction. To select the best revision from potential candidates we devised rule-set ranking measures, based on data mining measures and the novel concept of partial materialization. We evaluated our method with various configurations on both general-purpose and domain-specific KGs and observed significant improvements over a baseline Horn rule mining.

There are various directions for future work. First, we look into extracting evidence for or against exceptions from text and web corpora. Second, our framework can be enhanced by partial completeness assumptions for certain predicates (e.g., all countries are available in KG) or constants (e.g., knowledge about Barack Obama is complete). Finally, an overriding future direction is to extend our work to more complex nonmonotonic rules with higher-arity predicates, aggregates and disjunctions in rule heads.

Acknowledgments. We thank Thomas Eiter, Francesca A. Lisi and the anonymous reviewers for their constructive feedback about this work. The research was partially funded by the NWO VENI project 639.021.335.

References

1. Agrawal, R., Carey, M.J., Livny, M.: Concurrency control performance modeling: alternatives and implications. In: Performance of Concurrency Control Mechanisms in Centralized Database Systems, pp. 58–105 (1996)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)
3. Azevedo, P.J., Jorge, A.M.: Comparing rule measures for predictive association rules. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 510–517. Springer, Heidelberg (2007)
4. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka Jr., E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Proceedings of AAAI (2010)
5. Chen, Y., Goldberg, S., Wang, D.Z., Johri, S.S.: Ontological pathfinding: mining first-order knowledge from large knowledge bases. In: Proceedings of SIGMOD 2016, pp. 835–846 (2016)
6. Corapi, D., Russo, A., Lupu, E.: Inductive logic programming as abductive search. In: Proceedings of ICLP, pp. 54–63 (2010)
7. Darari, F., Nutt, W., Pirrò, G., Razniewski, S.: Completeness statements about RDF data sources and their use for query answering. In: Alani, H., et al. (eds.) ISWC 2013, Part I. LNCS, vol. 8218, pp. 66–83. Springer, Heidelberg (2013)
8. Erxleben, F., Günther, M., Krötzsch, M., Mendez, J., Vrandečić, D.: Introducing wikidata to the linked data web. In: Mika, P., et al. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 50–65. Springer, Heidelberg (2014)
9. Fierens, D., den Broeck, G.V., Renkens, J., Shterionov, D.S., Gutmann, B., Thon, I., Janssens, G., Raedt, L.D.: Inference and learning in probabilistic logic programs using weighted boolean formulas. TPLP **15**(3), 358–401 (2015)
10. Flach, P.A., Kakas, A.C.: Abduction and Induction: Essays on Their Relation and Integration. Applied Logic Series, vol. 18. Springer, Heidelberg (2000)
11. Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.M.: Fast rule mining in ontological knowledge bases with AMIE+. VLDB J. **24**, 707–730 (2015)
12. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proceedings of ICLP/SLP, pp. 1070–1080 (1988)
13. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min. Knowl. Discov. **8**(1), 53–87 (2004)
14. Józefowska, J., Lawrynowicz, A., Lukaszewski, T.: The role of semantics in mining frequent patterns from knowledge bases in description logics with rules. TPLP **10**(3), 251–289 (2010)
15. Katzouris, N., Artikis, A., Paliouras, G.: Incremental learning of event definitions with inductive logic programming. Mach. Learn. **100**(2–3), 555–585 (2015)
16. Lassila, O., Swick, R.R.: Resource description framework (RDF) model and syntax specification (1999)
17. Law, M., Russo, A., Broda, K.: Inductive learning of answer set programs. In: Fermé, E., Leite, J. (eds.) JELIA 2014. LNCS, vol. 8761, pp. 311–325. Springer, Heidelberg (2014)
18. Lehmann, J., Auer, S., Bühmann, L., Tramp, S.: Class expression learning for ontology engineering. J. Web Sem. **9**(1), 71–81 (2011)

19. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV system for knowledge representation and reasoning. ACM TOCL **7**(3), 499–562 (2006)
20. Lisi, F.A.: Inductive logic programming in databases: from datalog to DL+log. TPLP **10**(3), 331–359 (2010)
21. Lloyd, J.W.: Foundations of Logic Programming, 2nd edn. Springer, Heidelberg (1987)
22. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: a knowledge base from multilingual Wikipedias. In: Procedings of CIDR (2015)
23. Muggleton, S., Feng, C.: Efficient induction of logic programs. In: ALT, pp. 368–381 (1990)
24. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proc. IEEE **104**(1), 11–33 (2016)
25. Nickles, M., Mileo, A.: A hybrid approach to inference in probabilistic nonmonotonic logic programming. In: Proceedings of ICLP, pp. 57–68 (2015)
26. Ray, O.: Nonmonotonic abductive inductive learning. J. Appl. Logic **3**(7), 329–340 (2008)
27. Sakama, C.: Induction from answer sets in nonmonotonic logic programs. ACM Trans. Comput. Log. **6**(2), 203–231 (2005)
28. Taniar, D., Rahayu, W., Lee, V., Daly, O.: Exception rules in association rule mining. Appl. Math. Comput. **205**(2), 735–750 (2008)
29. Wang, Z., Li, J.: RDF2Rules: learning rules from RDF knowledge bases by mining frequent predicate cycles. CoRR abs/1512.07734 (2015)
30. Wrobel, S.: First order theory refinement. In: Raedt, L.D. (ed.) Advances in Inductive Logic Programming, pp. 14–33. IOS Press, Amsterdam (1996)