Learning Rules With Attributes and Relations in Knowledge Graphs

Pouya Ghiasnezhad Omran[1], Zhe Wang[2] and Kewen Wang[2]

[1] The Australian National University, Canberra, ACT, Australia
[2] Griffith University, Brisbane, QLD, Australia

Abstract
Rules capture high-level patterns in knowledge graphs and sometimes can provide virtual schemata for the data. Recent research has witnessed an increasing interest in learning rules automatically and applying the learned rules in knowledge graph inferences. While quite a number of rule learning systems have been developed, the formats of learned rules are still restricted, mostly resemble paths in the knowledge graphs. Such methods focus on the graph structure of the knowledge graph, assuming all the vertices and all the edges in the graph have an equal role (while only named or labelled differently). In this paper, we propose a method HARL (Hub Augmented Rule Learning) for learning rules containing attributes by treating certain binary predicates as unary predicates. A major component of HARL is an algorithm for identifying attributes for a given knowledge graph. HARL has been implemented through a scalable rule learner RLvLR. Our experimental results also show that HARL is scalable and outperforms RLvLR in most cases on major benchmark datasets.

Keywords
Rule Learning, Knowledge Graph, Knowledge Graph Completion

1. Introduction

Knowledge graphs (KGs) have proven to be useful for building interlinks between information sources by connecting entities of diverse types like people, universities, movies, animals, etc. This is particularly useful for information synthesis over multimodal media data, such as the fusion of medical documents consisting of X-ray images and medical reports (texts and tables). In artificial intelligence, a KG is usually a set of triples like $(Barty, \mathsf{bornInCity}, Brisbane)$, meaning that Barty was born in the city of Brisbane. Tools are available for retrieving such triples from various sources [1]. Thus, solutions for retrieving information can be obtained by technologies for KGs, such as efficient algorithms for link prediction. The objective of link prediction is to determine whether a pair of entities are connected via a relation. For instance, the link prediction task answers a query, $(Barty, \mathsf{bornInCity}, ?)$ that enquiries for a city name where Barty was born in. Existing methods for link prediction are mostly based on embedding based representation and graph neural networks (GNNs) [2], which adopt a black-box approach

In A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI 2022 Spring Symposium on Machine Learning and Knowledge Engineering for Hybrid Intelligence (AAAI-MAKE 2022), Stanford University, Palo Alto, California, USA, March 21–23, 2022.

EMAIL: p.g.omran@anu.edu.au (P. G. Omran); zhe.wang@griffith.edu.au (Z. Wang); k.wang@griffith.edu.au (K. Wang)

and are known to suffer from issues of explainability and flexibility. As a result, some researchers advocated providing explainable solutions to KG link prediction by learning rules. In particular, rules that contain constants are useful for both link prediction and reasoning. However, existing scalable rule learners based on embedding, e.g. [3, 4], can only learn the class of *closed path rules* (*CP-rules*) but are unable to learn rules containing constants. This assumption is vital since we face millions of entities in large KGs, which causes an explosion in the search space.

On the other hand, in the KGs, specific entities in specific predicates acts as attributes. For example, in FB15k, the entity USA in the second argument of the predicate $\mathsf{/film/film/country}(x, y)$ acts as a class that distinguish all films (x) that are produced in the USA and from those who are not. Such binary attribute reflects a reality of dominance of the USA film industry globally. We refer to such entity-predicates as hubs and model them through attribute facts. We propose a method to identify a set of hubs automatically. Consider an example from YAGO2 KG, we have $\mathsf{P : isCitizenOf\_E : Netherland}(x)$ as an attribute or hub. We can learn a number of patterns that cannot be expressed without such attributes, for instance, the rule $\mathsf{P : isCitizenOf\_E : Netherlands}(x) \wedge \mathsf{isCitizenOf}(x, y) \rightarrow \mathsf{livesIn}(x, y)$ would have only a much lower confidence degree if without the attribute $\mathsf{P : isCitizenOf\_E : Netherlands}(x)$ (three times less confidence degree than the rule including attribute). Thus, we will consider the more specific rules such as, $\mathsf{P : isCitizenOf\_E : Netherlands}(x) \wedge \mathsf{isCitizenOf}(x, y) \rightarrow \mathsf{livesIn}(x, y)$, instead of the more general rule, $\mathsf{isCitizenOf}(x, y) \rightarrow \mathsf{livesIn}(x, y)$.

In this paper, we propose a method HARL (Hub Augmented Rule Learning), for learning rules containing attributes. The idea is to regard such a rule as a closed path rule so that existing scalable rule learners can be employed. This is achieved by treating a binary predicate as a unary predicate, for example, an atom $\mathsf{/film/film/country}(x, USA)$ can be considered as an attribute $\mathsf{/film/film/country\_USA}(x)$. Thus, a major component of HARL is an algorithm for identifying attributes for a given KG. HARL can keep the search space in a manageable size while essential entities for learning rules are not missed. HARL has been implemented through a scalable rule learner RLvLR [4]. Our experimental results also show that HARL outperforms RLvLR in most cases on major benchmark datasets.

2. Preliminaries

A knowledge graph consists of a set of entities $\mathcal{E}$ as its vertices and its edges are directed and labelled with a set of predicates $\mathcal{P}$. An entity e is an object such as a place, a person, etc., and a fact (or link) is a triple (e, p, e'), which means that the entity e is related to another entity e' via the binary predicate p. Following the convention in knowledge representation, we denote such a fact as $p(e, e')$.

The format of rules we aim to learn extends closed-path rules (or CP rules), which is a language bias that is widely adopted in the rule learning literature for knowledge graphs; for instance, CP rules are the underlying formalism of Path Ranking Algorithms [5], RuleEmbedding [6], RDF2Rules [7] and ScaLeKB [3]. We extends CP rules with atoms about entity attributes.

An *attribute closed-path rule* (or an *ACP rule* or simply a *rule*) r is of the form

$$p_1(x_0, x_1) \wedge p_2(x_1, x_2) \wedge \cdots \wedge p_n(x_{n-1}, x_n) \wedge a_1(x_{l_1}) \wedge \cdots \wedge a_m(x_{l_m}) \rightarrow p(x_0, x_n). \quad (1)$$

Here each x_i $(0 \leq i \leq n)$ is a variable and $0 \leq l_j \leq n$ for each $1 \leq j \leq m$, each $p(u, v)$ is called an atom, and u and v are called respectively, the subject and object argument for p, and each $a(u)$ is an atom with a being a unary predicate. Intuitively, the rule r reads that if $p_1(x_0, x_1), p_2(x_1, x_2), \ldots, p_n(x_{n-1}, x_n), a_1(x_{l_1}), \ldots, a_m(x_{l_m})$ hold, then $p(x_0, x_n)$ holds too. The atom $p(x_0, x_n)$ is the head of r and the set of atoms $\{p_1(x_0, x_1), p_2(x_1, x_2), \ldots, p_n(x_{n-1}, x_n), a_1(x_{l_1}), \ldots . a_m(x_{l_m})\}$ is the body of r. The rule obtained from r by removing unary predicates in the body, that is, $p_1(x_0, x_1) \wedge p_2(x_1, x_2) \wedge \cdots \wedge p_n(x_{n-1}, x_n) \rightarrow p(x_0, x_n)$ is usually referred to as a *closed-path rule* or *CP rule* as the sequence of predicates in the rule body forms a path from the subject argument to the object argument of the head predicate.

To assess the quality of mined rules, we recall measures that are used in some major approaches to rule learning [3, 8]. Let r be a rule of the form (1). A pair of entities (e, e') satisfies the body of r, denoted $body(r)(e, e')$, if there is a way of substituting variables in $body(r)$ with entities in the KG such that (i) all atoms in $body(r)$ (after substitution) are facts in the KG, and (ii) x_0 and x_n are substituted with e and e' respectively. And (e, e') satisfies the head of r, denoted $p(e, e')$, if $p(e, e')$ is a fact in the KG. Then the support degree of r is defined as

$$supp(r) = \#(e, e') : body(r)(e, e') \wedge p(e, e')$$

That is, $supp(r)$ is defined as the number of entity pairs that satisfy both the head and the body of r.

The degrees of standard confidence (SC) and head coverage (HC) are defined as two forms of normalisation for $supp(r)$:

$$SC(r) = \frac{supp(r)}{\#(e, e') : body(r)(e, e')} \qquad HC(r) = \frac{supp(r)}{\#(e, e') : p(e, e')}$$

So, $SC(r)$ is the normalisation of $supp(r)$ through the number of entity pairs that satisfy the body, while $HC(r)$ is the normalisation of $supp(r)$ through the number of entity pairs that satisfy the head.

3. Learning Rules with Attributes

In this section, we present our approach to learn rules with attributes through a simple and effective method. We first describe how to automatically identify facts in the KG that express entity attributes , which we refer to as *attribute facts*, and then explain how to transform such attribute facts to a form that can be easily processed by existing rule learning methods.

3.1. Identifying Attribute Facts

As argued in [9, 10], the facts in existing KGs can be separated into (at least) two categories: those representing entity relations and those for entity attributes (hubs). For example, hasFather(Joe Biden, Joseph R. Biden) expresses the relationship between two entities Joe Biden and Joseph R. Biden, whereas gender(Joe Biden, Male) is naturally considered as an attribute of the entity Joe Biden. Sometimes it is not easy to draw a clear line between the two types of facts, but in general, entity relations often involve large numbers of entities, like Joe Biden and Joseph R. Biden, but the links between an attribute and entities are relatively sparse; in another word, entity attributes often involve a small number of entities to represent attribute values, such as the gender Male, which are linked to a huge number of other entities.

Existing rule learning approaches often do not distinguish entity attributes from entity relations as they can only learn the class of CP rules like

$$\mathsf{hasFriend}(x, y) \wedge \mathsf{gender}(y, z) \rightarrow \mathsf{gender}(x, z).$$

However, in many cases it makes more sense to consider 'gender' as an attribute when rules of the following more general form are learned.

$$\mathsf{hasParent}(x, y) \wedge \mathsf{gender}(y, \mathrm{Male}) \rightarrow \mathsf{hasFather}(x, y).$$

To identify attribute facts $p(e, e')$, we need to identify e' that can be used for attribute values, such as Male, and p that is suitable for representing attributes, such as gender. That is, we need to identify the attribute values e' and attribute predicates p.

For attribute values, we observe that they are often linked to a huge number of entities and thus form kind of "hubs" in the network. Based on the above discussion, such hubs can be identified by their connection densities.

We define the density of a hub, $den(e, p)$, as the (in and out) degrees of an entity e w.r.t. a predicate p as follows.

$$den_{in}(e,p) = \frac{\#e' : p(e',e)}{\#(e',e'') : p(e',e'')} \qquad den_{out}(e,p) = \frac{\#e' : p(e,e')}{\#(e',e'') : p(e',e'')} \tag{2}$$

In the definition of density, $\#e' : p(e, e')$ or $p(e', e)$ is the total number of occurrences of e (as a subject or object) in a fact of the KG, and it is normalized by the total number of facts about p in the KG.

Yet a hub e with high density is not necessarily suitable for an attribute value. Consider an extreme case where all the other entities e' are connected with e through a predicate p, then e is not an attribute value that can distinguish e' from other entities. From information theory, such an attribute does not provide any information gain as it does not express any distinguishing feature of associated entities. Thus, we need to select the hubs that both have high density and provide distinguishing features. We compute the entropy of hubs as follows.

$$ent_c(e,p) = -(d * log(d) + (1-d) * log(1-d)), \text{ where } c \in \{in, out\}, d = den_c(e,p) \tag{3}$$

Next, we want to identify those predicates that are suitable for attributes. This is based on the observation that such a predicate often associates an entity with its attribute values, and the number of possible values (like Male) is often far smaller than the number of entities that can be associated with the attribute (like Joe Biden). So we define the imbalance degree (imb) of a predicate p as follows.

$$imb_{in}(p) = \frac{\#e : p(e,e')}{(\#e' : p(e,e')) * |\mathcal{E}|} \qquad imb_{out}(p) = \frac{\#e : p(e',e)}{(\#e' : p(e',e)) * |\mathcal{E}|} \tag{4}$$

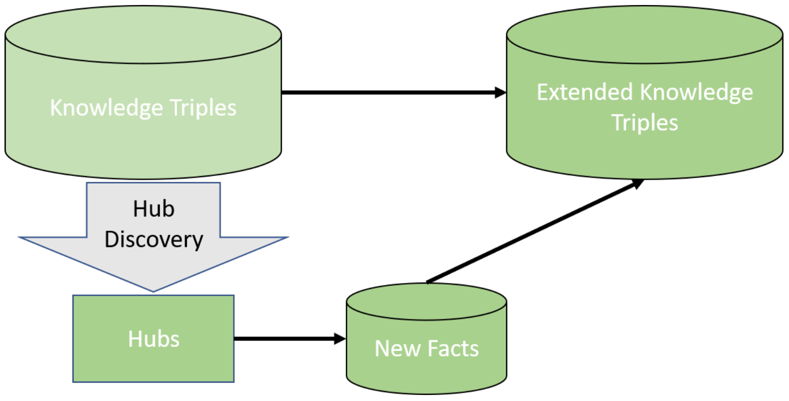

Figure 1: Enriching KG with hubs data flow.

Intuitively, the degree $imb_{in}(p)$ being large means that p has more potential subjects than objects, which suggests the objects of p could potentially be attribute values (whereas the subjects are more likely to be entities). In other words, the vertices with p being an in-coming edge are more likely to be attribute values than those with p being an out-going edge.

Finally, we combine the two measures to determine the likeliness of a fact with a hub e and a predicate p being an attribute fact.

$$att_c(e,p) = ent_c(e,p) * imb_c(p), \text{ where } c \in \{in, out\}. \tag{5}$$

The larger $att_{out}(e,p)$ (or $att_{in}(e,p)$) is the more likely a fact $p(e,e')$ (resp., $p(e',e)$) is an attribute fact, with e' being an entity and e representing its attribute value.

3.2. KGs with Attributes

Unlike existing embedding methods that embed attributes separately from the other relations [3, 11, 10], we propose a method to transform the attribute facts, so that existing rule learners like RLvLR can be easily adapted to learn rules with attributes from such facts. We illustrate the data flow of our data transformation in Figure 1, where we obtain an enriched KG through the identified attribute facts.

In general, attributes can be seen as unary predicates, whereas other relations are binary ones. For an attribute fact gender(Joe Biden, Male), it can be seen as a unary fact gender_Male(Joe Biden). Yet, such unary fact cannot be directly processed by existing rule learners as they only accept binary facts (i.e., triples) as inputs. Hence, we represent the unary fact gender_Male(Joe Biden) as a fact that is essentially a self-loop in the graph, i.e., gender_Male(Joe Biden, Joe Biden).

Formally, each attribute fact in the original KG of the form $p(e,e')$ (or $p(e',e)$) with e' being the attribute value can be transformed into a fact $P:p\_E:e'(e,e)$ (resp., $E:e'\_P:p(e,e)$), where $P:p\_E:e'$ (resp., $E:e'\_P:p$) is a new predicate. In Figure 2, we illustrate how attribute facts are transformed into self-loop in the KG.

The intuition of such a transformation is to represent attributes as self-loops in the KG, so that path-based rule learning method can be easily adapted to handle such a KG while the attribute

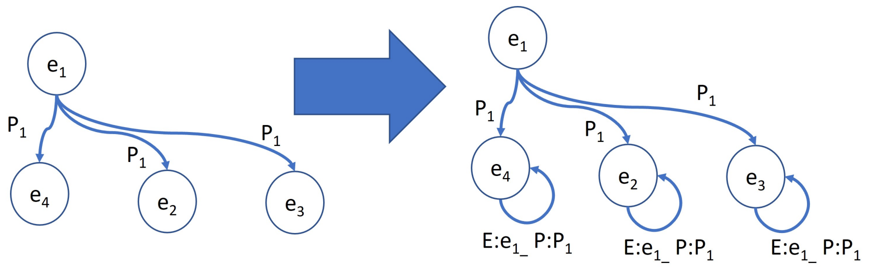

Figure 2: Example of enriching a KG with hubs.

atoms (as self-loops) can be conveniently identified in the learned rules. For example, a path $\mathsf{hasParent}(\text{Joe Biden}, \text{Joseph R. Biden}), \mathsf{gender}(\text{Joseph R. Biden}, \text{Male})$ is transformed into a path $\mathsf{hasParent}(\text{Joe Biden}, \text{Joseph R. Biden}), \mathsf{P : gender\_E : Male}(\text{Joseph R. Biden}, \text{Joseph R. Biden})$, which together with many other similar paths may induce a rule

$$\mathsf{hasParent}(x, y) \wedge \mathsf{P : gender\_E : Male}(y, y) \rightarrow \mathsf{hasFather}(x, y).$$

The rule can be rewritten as

$$\mathsf{hasParent}(x, y) \wedge \mathsf{P : gender\_E : Male}(y) \rightarrow \mathsf{hasFather}(x, y).$$

3.3. Learning Rules with Attributes

In Algorithm 1, we explain the complete process of identifying attribute facts, expanding the KG with transformed facts, and learning rules with attributes. Intuitively, we first select fact patterns $p(e, \cdot)$ or $p(\cdot, e)$ that can be considered attribute facts. We represent such a pattern as a triple (p, e, c) with $c \in \{in, out\}$, which should not be confused with a triple in the KG. We select such triples where the total number of facts based on such patterns that can be added to the KG is bounded by a expansion degree Exp. In what follows, we call such a triple a *hub*.

In line 1, we compute the frequency of all potential hubs. For each candidate hub (p, e, c), its frequency is the unnormalized density which is the numerator of Equation (2). To compute the frequencies, we just need to parse all facts in the KG once. We select top n candidate hubs, where $n = TopFreqNum$, to form the $Hublist$.

In lines 2 to 3, we compute the att measure for all potential hubs in $Hublist$ as we explained in Equation (5). Then, we sort the list of candidate hubs by their att values.

In lines 5 to 9, we consider the potential hubs one by one from the $Hublist$ with the highest att degree and create the corresponding new attribute predicate and facts. We stop this process when we hit the maximum expansion degree. The expansion degree is computed by the number of new facts divided by the number of facts in the original KG. The maximum expansion degree is given as input parameter, $MaxExp$. In line 10 we feed the new set of facts to RLvLR and obtain a set of attribute rules. In line 11 we return a set of rules with attributes.

Algorithm 1 Learn Rules with Attributes

Input: a KG as a set of facts $\mathcal{F}$
Parameter: a max expansion degree $MaxExp$ and an integer $TopFreqNum$
Output: a set of rules with attributes $\mathcal{R}$

```
1: $Hublist$ := selectTopFrequent($\mathcal{F}, TopFreqNum$)
2: computeAtt($Hublist$)
3: $Hublist$ := sortAtt($Hublist$)
4: $\mathcal{F}' = \mathcal{F}$
5: Let $NewFacts$ consist of attribute facts related to top hubs from $Hublist$
6: while $(|\mathcal{F}'| + |NewFacts| - |\mathcal{F}|)/|\mathcal{F}| \leq MaxExp$ do
7:     $\mathcal{F}' := \mathcal{F}' \cup NewFacts$
8:     Let $NewFacts$ consist of attribute facts related to top hubs from $Hublist$
9: end while
10: $\mathcal{R}$ := RLvLR($\mathcal{F}'$)
11: return $\mathcal{R}$
```

4. Context Aware Inference

There is a body of works for applying the learned rules to the known facts for inferring new facts in KGs. Most of these works focus on how the rules can be used in a restricted iterative way [12]. There is another stream of works that consider applying the uncertain rules once in the knowledge graph community [4, 13, 8].

In the latter stream of works, each rule is augmented with a degree of confidence and the inference module takes to the account these degrees to infer a new fact (which is also augmented with a degree of confidence). To control the quality and number of new inferred facts, these systems put a maximum threshold on the number of new facts and/or put a minimum threshold on the confidence of new facts.

The problem with this method is they do not consider the demand of KG regarding each new fact; they merely focus on the process which results in the new facts. For example, consider the predicate isParentOf. For each human person, usually, we have at most two entities as parents. In the case that we want to infer new facts regarding this predicate for a specific person, Ann, we consider all rules that result in the following partially grounded fact (isParentOf(x, Ann)). On the other hand, we need to consider how many instantiations of this fact are needed. If we have already two entities as parents of Ann, we should reject any more facts even if those facts are inferred via quite confident rules. For another entity, Pit, that we have no facts that state the parent of him, we should accept the new inferred fact even if the confidences of the corresponding rules are not high. Thus, in the context-aware inference method, we try to consider the confidence of the new fact production process and the demand of the new fact in the KG.

In the context-aware inference method, we will infer the new facts by considering the already related existing facts in the KG. We could obtain an average number of facts regarding each predicate and entity in a specific predicate argument. Later, when we want to infer new facts, if there are enough facts already in the KG, inferring new facts are likely to be overpredicting. To avoid overpredicting, we consider just the predictions which securely fill the gap (between the current number of facts and the average number). We give higher priority to the facts with higher SC if there are alternatives. So we sort all predictions regarding their SCs, and for each one $(p_1(e_1, e_2))$ we consider the gap between the average number of facts regarding specific entity and predicate and the concerning new fact. We obtain the Average Number of Facts (ANF) for each entity, predicate, and argument. Although this averaging method ignores the variations of the number of facts for a specific entity in the specific predicate, we propose it inferring a new fact in an extra cautious fashion. We do not claim the rejected new facts are not valid but based on this mechanism; we do not have strong support that the new fact is needed.

$$ANF_{p_1}^{Sub} = \frac{\#(e, e') : p_1(e, e')}{\#e : \exists y \, p_1(e, y)} \tag{6}$$

$$ANF_{p_1}^{Obj} = \frac{\#(e, e') : p_1(e, e')}{\#e' : \exists x \, p_1(x, e')} \tag{7}$$

$$((\#e' : p_1(e_1, e')) < ANF_{p_1}^{Sub}) \wedge ((\#e : p_1(e, e_2)) < ANF_{p_1}^{Obj}) \tag{8}$$

We accept a new fact if the condition (8) is satisfied. This condition demands the new fact populate the KG whence both subject and object have less than average relations regarding the predicate.

By this extra cautious method, we can avoid overpopulating KG by predicting too many facts w. r. t. any specific pair of subject-predicate or object-predicate with the cost of excluding many facts that might be valid. For example, in the case of nationality, the number of entities with the same nationality varies from a hundred thousand to a billion based on real-world data. By considering the average, we might exclude many new valid facts for a more populated country. Hence, the excluded facts by this mechanism are not wrong, but if we have a high priority to infer fewer facts, we exclude them.

5. Experiments

Based on the methods presented in previous sections, we have implemented Hub Augmented Rule Learning (HARL) and conducted three sets of experiments to evaluate the new system.

The first set of experiments aim to evaluate the effect of new hub predicates on rule learning. In the second set of experiments, we use the learned rules to infer new facts via the original inference module of RLvLR and the new context aware inference module. The results show the usefulness of hub-augmented rules to infer more quality facts. The third set of experiments aim to evaluate the improvements of RLvLR on link prediction task regarding using hubs as new predicates.

In Table 1, we show the statistics of three widely used Knowledge graphs. We randomly selected 50 target predicates for target predicates, and for YAGO2 core, we used all 32 predicates as target predicates. All results shown in the following experiments are average results for the corresponding target predicates.

Table 1
Benchmark KG specifications

KG	# Facts	# Entities	# Predicates
FB15K	542K	15K	1345
FB15KSE	272K	15K	237
YAGO2 core	948K	470K	32

To show the advantage of identifying the hubs and populating the KG with hubs for rule learning. We introduce a new kind of rule redundancy elimination, the Concise Rule Set.

5.1. Eliminating Redundant Rules: Concise Rule Set

Rule learners can be evaluated by the number of distinct quality rules that they can learn. AMIE+ [8] eliminates duplicate rules. In this process, two rules are considered duplicated only if they have the same head predicate, the same number of atoms (body length), and the same head coverage (or support). While such elimination merely considers rules with same syntax as the same rule (e.g. $p_3(x, y) \wedge p_2(x, y) \rightarrow p_1(x, y).$ and $p_2(x, y) \wedge p_3(x, y) \rightarrow p_1(x, y).$), it does not consider the semantic similarity like, $\mathsf{livesIn}(x, y) \rightarrow \mathsf{isCitizenOf}(x, y).$ and $\mathsf{livesIn}(x, y) \wedge \mathsf{Person}(x) \rightarrow \mathsf{isCitizenOf}(x, y)$. Intuitively the last second atom of second rule is obvious and does not add any information to the rule while it makes a new (syntactically) rule. These kind of semantically redundancy can discredit the evaluation of rule learning systems based on the number of quality rules. To overcome this challenge we introduce a novel method to obtain a set of rule with minimum redundancy.

We deployed a greedy search to find a set of rules with minimum redundancy w.r.t. fact inferences. Each rule r is associated with a set of facts that can be inferred from the rule, denoted $fact(r)$. For a set of rules $\mathcal{R}$, $fact(\mathcal{R}) = \bigcup_{r\in\mathcal{R}} fact(r)$. To obtain a concise set of rules, we start with an empty set $\mathcal{R}$ and add rules into $\mathcal{R}$ one by one from the set of all learned rules, following the descending order of the confidences of the rules (i.e., their SCs). Before adding a rule r to $\mathcal{R}$, we first check if the facts inferred by r can already be inferred by those in $\mathcal{R}$. We define the novelty degree for each rule r as follows:

$$\mathsf{noveltyDegree}(r) = \frac{|fact(r)| - |fact(r) \cap fact(\mathcal{R}))|}{|fact(\mathcal{R}))|}$$

We add the rule r to $\mathcal{R}$ only if the novelty degree is higher than a threshold.

5.2. Learning Rules with Attributes

Table 2 shows the average numbers of rules ($\#R$) with the minimum quality SC≥ 0.1 and HC≥ 0.01 as these values are used in RLvLR [4] and AMIE+ [8]. $\#QR$ indicates the average number of rules with the following minimum qualities, SC≥ 0.8 and HC≥ 0.01. $\#CR$ indicates the average size of the concise rule sets (obtained from all $\#R$ learned rules). $\#QCR$ indicates the average size of the quality concise rule sets (obtained by from $\#QR$ high quality rules). $\#CR-H$ indicates the average number of concise rules that contains at least one hub predicate (obtained from the concise rule set of size $\#CR$).

Table 2
Performance of HARLand RLvLR on Rule Learning task

Benchmark	HARL					RLvLR				
	#R	#QR	#CR	#QCR	#CR-H	#R	#QR	#CR	#QCR	#CR-H
FB15K	**267**	**123**	**190**	**48**	66	153	15	117	10	0
FB15KSE	**252**	**80**	**205**	**34**	56	171	21	165	17	0
YAGO2core	**70**	**32**	**52**	**14**	13	33	4	32	4	0

Table 3
Performance of HARL and RLvLR on Facts Inference task

Benchmark	HARL			RLvLR		
	#UNF	#NF	#QNF	#UNF	#NF	#QNF
FB15K	66152	**578**	**314**	**66215**	455	276
FB15KSE	**56245**	**815**	**293**	55952	627	216
YAGO2core	**182676**	**2053**	**279**	175379	1983	265

Table 4
Performance of HARL and RLvLR running characteristics

Benchmark	HARL				RLvLR
	Time	#Hub	EXP	Minatt	Time
FB15K	389	73	0.09767	0.00138	310
FB15KSE	299	55	0.09118	0.0014	276
YAGO2core	403	255	0.09889	0.0000003	292

Compared to RLvLR, HARL showed significantly better performance in terms of both the rules with ordinary filtering and concise filtering. The superiority of HARL is more obvious in learning high quality rules, that is, among all the rules learned, HARL has a consistently higher number of high quality rules.

5.3. Inferring New Facts via Learnt Rules

To estimate the predictive power of the corpus of learned rules, we calculated the number of facts can be predicated by applying learned rules on the given facts via RLvLR inference module and context-aware inference module. To aggregate the different rules with same target predicate we use Noisy-OR similar to RLvLR [13] and AMIE+ [8]. Table 3 shows the numbers of RLvLR inference predicted facts (#UNF), those predictions via context-aware inference facts (#NF), and those context-aware inferred facts with CD≥ 0.8 (#QNF).

The running of two systems have the following characteristics regarding time and other features 4. #Hub indicates the number of hubs that we consider before the number of new facts hits the max expansion degree ($MaxExp$ in algorithm 1), which we assigned 0.1 for max expansion degree in these experiments. EXP is the actual expansion degree. Minatt is the att value of the last hub that we consider.

Table 5
Performance of HARL and RLvLR on Link Prediction task

Benchmark	HARL				RLvLR			
	MRR	MR	H1	H10	MRR	MR	H1	H10
FB15K	**0.36**	**169**	**0.23**	**0.58**	0.3	207	0.2	0.48
FB15KSE	0.21	**1171**	0.14	0.34	0.21	1180	**0.15**	0.34
YAGO2core	0.06	**332820**	0.04	**0.1**	0.06	333778	0.04	0.09

5.4. Link Prediction via Learnt Rules

We additionally evaluated the quality of rules through link prediction. We consider two queries, $p(e, ?)$ and $p(?, e')$, for each fact, $p(e, e')$, in the testing data in each benchmark. For FB15k and FB15KSE the separation of test and train data has been done, while for YAGO2core, there is no previously prepared data, so we randomly separate 90% for train and 10% for the test. We applied learnt rules on the training data to predict the missing entities, and the testing data verified the correctness of such predictions. The predictions were measured using Mean Rank (MR) in which the less value indicates better performance and Mean Reciprocal Rank (MRR) in which the higher value indicates better performance. We also deployed Hits@1 and Hits@10. These four quality measurements are commonly used in the Knowledge Graph Completion literature [2]. In particular, we first ranked each inferred fact based on the number of rules inferring it. MR is the average number of ranks. MRR is the average number of the reciprocal ranks of correctly predicated entities. Hits@10 and Hits@10 are the proportion of correctly predicted entities that are ranked among the top ten and one, respectively. Table 5 summaries the results, where H10 and H1 are the Hits@10 and Hits@1 scores.

HARL outperforms RLvLR in FB15K in all quality measures significantly while in other two benchmark it performs in par. Since the FB15kSE and YAGO2core are more difficult tasks regarding link prediction, it might be the case that proposing the hub could not help the link prediction.

To see how the hubs can effect the mined rules consider the following two rules from YAGO2core.

$$SC : 0.03, HC : 0.07, \mathsf{wasBornIn}(x, z) \wedge \mathsf{isLocatedIn}(z, y) \rightarrow \mathsf{isCitizenOf}(x, y).$$

$$SC : 0.61, HC : 0.01, \mathsf{P} :< \mathsf{graduatedFrom} > \mathsf{\_E} :< \mathsf{University\_College\_Dublin} >(x)$$
$$\wedge \mathsf{wasBornIn}(x, z) \wedge \mathsf{isLocatedIn}(z, y) \rightarrow \mathsf{isCitizenOf}(x, y).$$

The standard confidence of the second rule improved by about 20 times. Since we add a condition at the body of the rule, we make it more specified, so it fires less number of facts, but those shots are more accurate and improve the SC significantly. On the other hand, since the number of resulting facts in the second rule decreased, the head coverage also reduced, making the rule less general. While, the first rule is filtered due to its low SC, the specified version of it has higher SC and qualified to be in the output set of rules.

From FB15k KG, the following hubs are created as they have top att.

1. $\mathsf{P} :< \mathsf{/people/person/gender} > \mathsf{\_E} :< \mathsf{male} >(x)$
2. $\mathsf{P} :< \mathsf{/people/person/spouse\_s./people/marriage/type\_of\_union} > \mathsf{\_E} :< \mathsf{matrimony} >(x)$

3. E :$<$ matrimony $>$ _P :$<$ /people/marriage _union_type/unions_of_this_type./people/marriage/spouse $>(x)$
4. P :$<$ /people/person/nationality $>$ _E :$<$ USA $>(x)$
5. P :$<$ /people/person/profession $>$ _E :$<$ actors $>(x)$
6. E :$<$ actors $>$ _P :$<$ /people/profession/people_with_this_profession $>(x)$
7. P :$<$ /film/film/language $>$ _E :$<$ eng $>(x)$
8. P :$<$ /film/film/country $>$ _E :$<$ USA $>(x)$
9. P :$<$ /film/film/release_date_s./film/film_regional_release_date /film_release_distribution_medium $>$ _E :$<$ DVD $>(x)$
10. P :$<$ /film/film/release_date_s./film/film_regional_release_date /film_release_region $>$ _E :$<$ USA $>(x)$

From this example, we can observe the identified hubs in FB15k are intuitive and shows which binary attributes can separate the related entities into two sets. In the real case scenario, regarding the film industry, we have some dominant categories that can distinguish the entities, like the movies produced in the USA in contrast with those that were produced in any country other than the US. With our proposed modelling, we can reflect on this disparity between different values of one attribute.

6. Related Works

Several rule learning systems, including AMIE+ [8], RuDiK [14] and anyBURL [15], allow a set of entities to be considered in the partially grounded rules. Gu et al [16] introduce a language bias for their rule learning method, in which the head atom and last atom of the rule's body are partially grounded but no entities are allowed in other atoms. However, they treat all entities equally in the partially grounded rules that are leaned. These method do not have a mechanism for selecting a set of entity-predicate pairs for attribute values. Consequently, the search space of possible rules grows vastly and these systems struggle to scale to the real-world massive KGs.

The other related stream of works is attribute knowledge graph modelling. In general, our work is different from this line of works for at least two reasons: (1) they assume an input KG contain attributes while we create attributes out of the given KG; (2) the nature of attributes they consider is only numeric or categorical, while ours is a particular case of categorical ones.

The attribute KGs differ in terms of the nature of attributes. [11] and [10] can model KGs that contain numeric feature vectors as attributes of nodes (entities). In [11], authors consider a text related to an entity like the title of a paper as a feature of paper node. They use methods in Natural Language Processing to construct a feature vector for representing the title feature in a numeric representation rather a string of characters.

In [9], authors distinguish the relations from attributes and consider only categorical attributes. The attribute KG that they consider, FB24K, is extracted from Freebase. The number of values of attributes range from 1 to 877 categorical attributes and the number of attributes is 314. While what an attribute in their approach is similar to the hubs in our method, our attributes are binary. Furthermore, we do not remove the original facts, and an entity can act as both an attribute value and an entity. For example, the USA, in regards to predicate nationality, acts as an attribute value. At the same time, it acts as an entity in the predicate neighbour. However,

[9] does not consider the attribute values as entities, so ignore the dual role of a thing as an entity and attribute/value.

7. Conclusion

In this paper, we proposed an approach to learn attributed rules over KGs. In contrast to existing attribute KG modelling approaches that assume attributes as input, we can mine the attributed rules from KG without an explicit set of attributes. We achieved this by proposing a binary attribute that can easily be embedded into the closed path rules and an identifying mechanism that automatically mines such attributes from KG. We enhance the original KG with new predicates (hubs) and corresponding facts. Then we feed the improved KG to a state-of-the-art rule learner, RLvLR, to learn attributed rules. We also proposed a novel inferring mechanism that considers the demands of a new inferred fact and the quality of the rules that infer it. Finally, we evaluated our system HARL on widely used benchmarks in rule learning and link prediction. Our experimental results demonstrate that HARL improved RLvLR in terms of both the number of mined quality rules, inferred facts, and accuracy in link prediction.

References

[1] T. Pellissier-Tanon, G. Weikum, F. Suchanek, YAGO 4: A Reason-able Knowledge Base, in: ESWC, 2020.

[2] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, Z. Duan, Knowledge graph completion: A review, IEEE Access 8 (2020) 192435–192456.

[3] Y. Chen, D. Z. Wang, S. Goldberg, ScaLeKB: scalable learning and inference over large knowledge bases, The International Journal on Very Large Data Bases 25 (2016) 893–918.

[4] P. G. Omran, K. Wang, Z. Wang, Scalable rule learning via learning representation, in: IJCAI, 2018.

[5] M. Gardner, T. Mitchell, Efficient and Expressive Knowledge Base Completion Using Subgraph Feature Extraction, in: Conference on Empirical Methods on Natural Language Processing, September, 2015, pp. 1488–1498.

[6] B. Yang, W.-t. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, in: ICLR, 2015, p. 12.

[7] Z. Wang, J.-Z. Li, RDF2Rules: Learning Rules from RDF Knowledge Bases by Mining Frequent Predicate Cycles., arXiv preprint (2015).

[8] L. Galárraga, C. Teflioudi, K. Hose, F. M. Suchanek, Fast rule mining in ontological knowledge bases with AMIE+, The International Journal on Very Large Data Bases 24 (2015) 707–730.

[9] Y. Lin, Z. Liu, M. Sun, Knowledge representation learning with entities, attributes and relations, in: IJCAI, 2016.

[10] W. Liu, H. Cai, X. Cheng, S. Xie, Y. Yu, H. Zhang, Learning High-order Structural and Attribute information by Knowledge Graph Attention Networks for Enhancing Knowledge Graph Embedding, arXiv preprint (2019).

[11] Z. Hu, Y. Dong, K. Wang, K. W. Chang, Y. Sun, GPT-GNN: Generative Pre-Training of Graph Neural Networks, in: SIGKDD, ACM, 2020, pp. 1857–1867.

[12] M. Svato, S. Schockaert, J. Davis, STRiKE : Rule-Driven Relational Learning Using Stratified k-Entailment, in: ECAI, 2020.

[13] P. G. Omran, K. Wang, Z. Wang, An Embedding-based Approach to Rule Learning in Knowledge Graphs, TKDE 33 (2021) 1348—-1359.

[14] S. Ortona, V. V. Meduri, P. Papotti, Robust discovery of positive and negative rules in knowledge bases, in: International Conference on Data Engineering (ICDE), IEEE, 2018, pp. 1180–1191.

[15] C. Meilicke, M. W. Chekol, D. Ruffinelli, H. Stuckenschmidt, Anytime bottom-up rule learning for knowledge graph completion, in: IJCAI, 2019.

[16] Y. Gu, Y. Guan, P. Missier, Towards Learning Instantiated Logical Rules from Knowledge Graphs, arXiv preprint (2020).