# Rule Learning

**Hong Wu, Kewen Wang**

College of Intelligence and Computing, Tianjin University, China

wh0625@tju.edu.cn, k.wang@griffith.edu.au

## Abstract

Our work focuses on mining closed-path rules of first order rules from large Knowledge Graphs (KGs). Based on the previous related work, we propose an improved rule learning system with a hierarchical sampling algorithm and a simplified method of rule search which greatly contributed to the scalability of our approach. Experiments are conducted on three classic benchmarks of deterministic KGs compared with the latest rule learning systems RLvLR and AMIE+ via the task of rule mining. The results show that the improvements make the system more scalable, can handle large KGs more efficiently and mine more quality rules with a longer length.

## 1 Introduction

With the popularity of *Knowledge Garphs (KGs)*, it is widely uesd in various fields, suh as intelligent web search, QA, advertising recommendation, social media relationship mining, biomedicine, etc. However, the huge amount of data makes it more difficult to construct or complete knowledge graphs manually or automatically. How to efficiently process large-scale KGs which contain abundant information and great value, and obtain valuable knowledge related to concerns, are the important issues to be solved urgently.

Typically, knowledge garphs are constructed based on RDF triples. However, the sparse or dense facts surrounding an entity or a relationship are not sufficient to embody the pattern or amount of information in the context of big data. Rules are different from the triple form of KGs. It is a higher level abstract pattern representation. The laws they describe can help us better understand the data. It can make use of the new facts generated by rules to further complete the KGs.Also, it can detect the hidden faults in KBs, or be used to reasoning and so on.

Therefore, as an upstream research problem in the field of knowledge graphs applications, rule learning has certain research and practical significance.

### 1.1 Knowledge Graphs

Knowledge graph $K = (E, P, F)$ is a structured semantic knowledge base, which describes concepts and their mutual relationships in the physical world with a symbolic form, where $E, P, F$ are the sets of entities, predicates, and facts respectively. The basic unit is the triple of fact $f : (e_s, p, e_o)$ which denotes subject entity $e_s$ is related to another object entity $e_o$ by the binary predicate $P$. Entities are connected by relationships to form a network of knowledge structures. In order to facilitate the latter rule representation, we convert a fact representation into $p(e_s, e_o)$ for convenience. There are many KGs for research and business like YAGO [Suchanek *et al.*, 2007], DBpedia [Auer *et al.*, 2007], Wikidata [Vrandečić and Krötzsch, 2014] and Freebase [Paritosh *et al.*, 2008]. The huge amount of data, incompleteness, and noise data are all difficulties in its research.

### 1.2 Representation Learning

The entities, concepts and relationships in knowledge graphs are represented by discrete and explicit symbols, which are difficult to be directly applied to deep learning models such as neural networks based on continuous numerical representation. In order to effectively utilize the symbolic knowledge in the KGs, researchers have proposed a large number of representation methods for knowledge graphs. These methods transform the entities and predicates of discrete KGs into continuous dimensional space, such as **RESCAL** [Nickel *et al.*, 2012], **TransE** [Antoine *et al.*, 2013], **PTransE** [Lin *et al.*, 2015] and **DistMult** [Yang *et al.*, 2015]. Deterministic KG embeddings have been extensively explored by recent work. In [Chen *et al.*, 2018], they propose a novel uncertain KG embedding model **UKGE**, which aims to preserve both structural and uncertainty information of relation facts in the embedding space.

Based on these embedded representation models, a more reasonable *"rule representation"* model can be studied and a rule mining system capable of handling large KG is designed.

### 1.3 Rule Mining

**Rule**

In this paper, we aim to mine the closed-path rules (CP rules) of the Horn rules. A Horn rule is composed of a head and a body, where the head is a single atom and the body is a set of atoms. An atom is a fact which has variables at the position of subject and object. In Horn rules, CP rules are of the form:

$$P_1(x, z_1) \wedge P_2(z_1, z_2)... \wedge P_n(z_{n-1}, y) \rightarrow P_t(x, y). \quad (1)$$

Here $P_t$ is head predicate, and the other are body predicates. The variables $x, y, z_i$ can be instantiated by entities. For example: $BornIn(x, z) \land Country(z, y) \rightarrow Nationality(x, y)$, where the lenth of rule is set to 2 ( not include the head atom). It should be noted that although the CP rule contains the symbol "$\rightarrow$", it has a different meaning from the implication symbol in the *First-Order Logic*. We can judge a rule is valid by the head atom of an instantiated rule holds if all body atoms of the instantiated rule appear in the KB.

This form of rules is called closed-path because the sequence of predicates' variables in the body of rule forms a closed path from the subject argument to the object of the head target predicate. Due to the serial path, we factor in the inverse of predicates in order to mine more rules in practice. We also allow recursion in the CP rules, which means the predicate of head can appear in the body part.

**Inductive Logic Program**

Early research on generating CP rules in KBs focused on *Inductive Logic Programming (ILP)* [Stephen, 1997], which mine rules by exploring the space of all possible hypotheses (rules). Although ILP is a mature field, it is still difficult to mine logic rules from the knowledge bases. On one hand, KBs are based on the *Open World Assumption (OWA)*, which means that the unknown information cannot be used as negative samples. On the other hand, the scalability of these methods are insufficient to handle the large amount of data contained in large KGs. The **AMIE** [Galárraga *et al.*, 2013] system based on ILP uses the partial hypothesis of the *Partial Completeness Assumption (PCA)* to "guess" the counterexamples for mining rules from the KBs, and the optimized **AMIE+** [Galárraga *et al.*, 2015] proposes strategies such as pruning and approximation to enable the system to exploring the search space more effectively. In addition, **Neural LP** [Yang *et al.*, 2017], **ScaLeKB** [Chen *et al.*, 2016], **SWARM** [Barati *et al.*, 2016] and other systems have been proposed, but in dealing with large-scale KBs, the efficiency of systems and the number or quality of rules are still the main challenges of the existing systems.

**The Rule Learner Based on Embedding**

[Yang *et al.*, 2015] based on the knowledge graphs embedding representation model **DistMult**, the two tasks of link prediction and rule mining are compared with other work for the first time. In the task of rule mining, the score function is calculated by using the predicate matrix operational similarity as the criterion for selecting rules, and the candidate rule queue is obtained. In the rule extraction task, it is only compared with *AMIE*, and only applied to the small data set FB15K-401, which is slightly better than *AMIE* in terms of mining efficiency and number of rules.

The latest work **RLvLR** system [Omran *et al.*, 2018] is proposed to solve the scalability of processing large knowledge graphs. It adopts a new sampling algorithm combined with the embedded representation learning model and two scoring functions: *Synonymy* and *Co-cocurrency*. It is superior to the current optimal **AMIE+** system, and can mine more high-quality CP rules efficiently. But there are still some improvements in the design of algorithm. When the

---

**Algorithm 1** Learn rules for a KG and a target predicate

**Input**: a KG $K = \{E, P, F\}$, a predicate $P_t$
**Parameter**: an integer $MaxLen \geq 2$, and rule degree $DEGREE = \{R_{minSC}, R_{minHC}, QR_{minSC}, QR_{minHC}\}$ and parameters list of embedding $EmbParas$
**Output**: a set $Rule$ of CP rules

1: $E_0, P_0, F_0 = \mathsf{SampleByPt}(K, P_t)$
2: **for** $2 \leq len \leq MaxLen$ **do**
3:     $E_i, P_i, F_i = \mathsf{SampleByLength}(E_0, len)$
4:     $K^{'} = \{\cup_{i=0}^{n-1} E_i, \cup_{i=0}^{n-1} P_i, \cup_{i=0}^{n-1} F_i\}$
5:     $Ent_{emb}, Pre_{emb} = \mathsf{Embedding}(K^{'}, EmbParas)$
6:     $Candidate = \emptyset$
7:     $Result = \mathsf{RuleSearch}(K^{'}, P_t, Ent_{emb}, Pre_{emb}, len)$
8:     Add $Result$ to $Candidate$
9: **end for**
10: $Rule = \mathsf{EvaluateAndFilter}(K, Candidate, DEGREE)$
11: **return** $Rule$

---

cardinality of data or the number of predicates are in large-scale and the length of rule becomes longer, the search space is disastrously huge. In addition to this, matrix operation is too slow in the evaluation phase, which must reduce the running speed of the system to a certain extent.

## 2 The Framework of Algorithm

As shown in Algorithm 1, our algorithm framework has several improvements based on the **RLvLR** system.

For a certain knowledge graph, we randomly select a predicate as the target predicate $P_t$ to mine as many high-quality rules as possible with body's length (the number of predicates in the body part, excluding the head) less than $MaxLen$. The $Degree$ contains two types of metric thresholds for the quality rules $R$ and high quality rules $QR$: $HC$ and $SC$. For distinct embedding models, there are different parameters $EmbParas$.

### 2.1 Sample

Due to the characteristic of closed-path rules, the path of body atoms is equivalent to the sequence of predicates. We can apply the $\mathsf{Sampling}(\cdot)$ algorithm to get a smaller KB $K^{'} = \{E^{'}, P^{'}, F^{'}\}$ that contains all the potential predicates with related entities and facts before working directly on large KB. Further improvements to the sampling algorithm have led to a significant increase of scalability, so that longer CP rules that RLvLR cannot process can be mined in our system.

As can be seen from Figure 1, begining with the target predicate $P_t$ (get $P_0$), the related entities are sampled (get $E_0$). Then the sample range of predicates is expanded (get $P_i$) according to the maximum length of the body atoms to obtain the related entities and facts (get $E_i$ and $F_i$). The incrementing of $i$ is started by a non-repeating entity $e \in E_{i-1}$, and the predicate $P$ having a certain relationship $P(e, e^{'})$ with $e$ is sampled into the candidate $P_i$ set.

Assuming the $MaxLen = n$, the incrementing process of the sample is as follows:

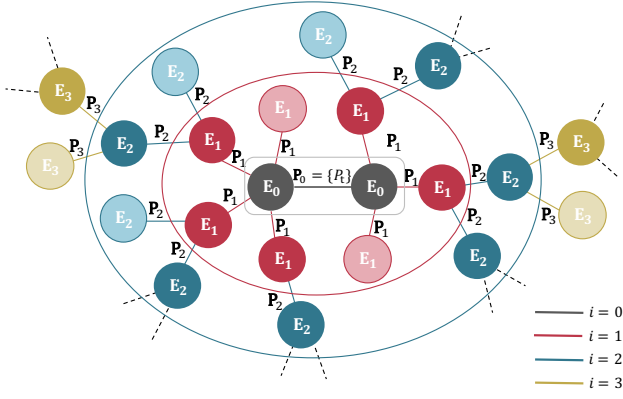Figure 1: Sample for the target predicate $P_t$, where $i = 3$. Different colors represent the phases of expansion. The light-colored entities denotes duplicate ones and no longer expand.

- $K_0 = \{E_0, P_0, F_0\}$, where $P_0 = \{P_t\}$,
  $E_0 = \{e | P_t(e, e^{'}) \text{ or } P_t(e^{'}, e) \in F\}$,
  $F_0 = \{P_t(e, e^{'}) | P_t(e, e^{'}) \in F\}$.

- $K_i = \{E_i, P_i, F_i\}, 0 < i < n$ (when $i < 2, E^{'}_{i-2} = \emptyset$),
  **for** $f = P(e, e^{'}) \in F$ **or** $f = P(e^{'}, e) \in F$ :
    **if** $e \in E_{i-1} - E^{'}_{i-2}$:
      **add** $e^{'}$ **into** $E_i$, **add** $P$ **into** $P_i$, **and add** $f$ **into** $F_i$.

The final sampled $K^{'} = \{E^{'}, P^{'}, F^{'}\}$ is a subset of K, where $E^{'} = \cup_{i=0}^{n-1} E_i$, $P^{'} = \cup_{i=0}^{n-1} P_i$ and $F^{'} = \cup_{i=0}^{n-1} F_i$.

**Reduce the candidate set of predicate**
Although there must be duplicate predicates in the set $P_i$ at each $i$th layer, it doesn't matter because we allow recursion the CP rules. But why we still remove duplicate entities in sample phase?

The expansion from $E_0$ to $E_i$ is to maintain the feature of CP rules: $P^s_1 = P^s_t$, but it always diverges outward without forming a closed loop, i.e. $P^o_n = P^o_t$, which only increases the range of sampling in vain, resulting in extremely increasing the number of candidate predicates. Finally, it will take lots of time and space complexity to search for predicate sequences which will affect the efficiency of system. In the work of repeating the **RLvLR**, they first sample all the entities and predicates of maximum layers, and then perform rule search. In most cases, when the body length reaches 4, $P^{'} = P$, causing the effect of sampling invalid and memory overflow. In summary, it removes the outward diverging path and formed a closed sequence of predicates, and ensures that loop caused by repeated occurrences of several connected predicates is avoided in the sampled graph.

In the latter rule search phase, as shown in Figure 2, the sequence obtained by performing the Cartesian product operation on the predicate sets $P_i$ of each layer can effectively reduce the computation cost compared to using the Cartesian product of all the predicate set $P^{'}$. The approach of obtaining the candidate sets of predicate fundamentally contributes to the scalability of our approach to deal with the rules whose body atoms are greater than 4.
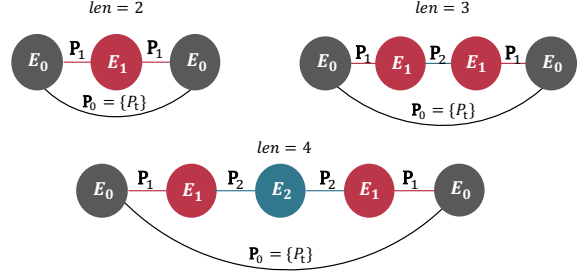


Figure 2: The predicate sets after sampling contain the predicates on the sequence of path.

## 2.2 Embedding Model

After sampling, entities and predicates in $K^{'}$ are embedded into the representation of vector ($ent_{emb}$ and $pre_{emb}$) using the embedding model in Embedding($\cdot$), respectively. The phase is prepared for calculating score functions in rule search stage. Unlike **RESCAL**, which is used in **RLvLR** to embed predicates as matrices, we use those such that **TransE** can project predicates into vectors.

The embedding model represents entities and relationships in the knowledge graph, defining a scoring function to learn the continuous representation of entities and relationships. Among the various models, the Translational Distance Model is a outstanding and relatively simple but effective kind of model. Given a fact: $P_r(e_h, e_t)$, the **TransE** model represents the relationship as translation vector $\vec{r}$, and the scoring function is defined as: the distance between $\vec{h} + \vec{r}$ and $\vec{t}$:

$$f_r(h, t) = -||\vec{h} + \vec{r} - \vec{t}||_{1/2} \qquad (2)$$

When this fact exists in KG, the score $f_r(h, t)$ is higher, i.e. $\vec{h} + \vec{r} \approx \vec{t}$.

The choice of embedding models needs to be considered in conjunction with the methods of calculating the score functions. In the next phase Rule Search, it will be further explained why we choose **TransE**.

## 2.3 Simplified Rule Search

In the smaller KG $K^{'}$, the task of searching for candidate rules in RuleSearch($\cdot$) is actually reduced to the search for plausible paths of body predicates. As shown in formula 1, the target predicate of the head has been fixed. With the set of hierarchical predicates obtained by the sampling algorithm, the range of the search predicate sequence is reduced, and the candidate sequence needs to be filtered by the score function to finally obtain the candidate. We use the vector obtained from the previous embedding model to further simplify the score function. Score functions are needed to measure which paths conform to both the form of cp rules and the intrinsic meaning of the rules and further prune for the search space to handle large KGs more efficiently.

Let's take the example of a rule with $length = 3$, where the predicates of head and body are included:

$$P_1(x, z) \wedge P_2(z, y) \rightarrow P_t(x, y).$$

### Score Functions

**Synonymy Score Function.** Before we introduce synonymy, we should abstract out the form of CP rules first. The path of predicates in body can be abstracted into a compositional relation $P_{body}$ that approximates target predicate $P_t$ if the rule is reasonable. Therefore, *synonymy* refers to this similar relationship between $P_{body}$ and $P_t$. In **RLvLR** and the work of [Yang *et al.*, 2015], the embedding of compositional relation path is defined as the matrix multiplication of predicates. Here we consider the characteristics of **TransE** and combine the previous methods to compute the synonymy:

$$f_{syn}(r) = sim(P_{body}, P_t) \qquad (3)$$
$$= sim(\mathbf{V}_{P_1} + \mathbf{V}_{P_2}, \mathbf{V}_{P_t})$$
$$= exp(-||((\mathbf{V}_{P_1} + \mathbf{V}_{P_2} - \mathbf{V}_{P_t}||_2)$$

The geometric meaning of vector addition is to use the Parallelogram Law, and the final sum vector is obtained from the beginning of the first vector to the end of the last vector, which is a more theoretical representation for the embedding of body's path.

Compared to matrix multiplication which has a relatively large amount of calculation with the increase of rule length, the addition of vector itself reduces the complexity of calculation and satisfies the addition commutative law, so the results of the same predicates in different order combinations are the same. For two predicates $P_1$ and $P_2$, we only need to calculate $\mathbf{V}_{P_1} + \mathbf{V}_{P_2}$ once, and for matrix multiplication in **RLvLR**, they need to calculate $\mathbf{M}_{P_1} \cdot \mathbf{M}_{P_2}$ and $\mathbf{M}_{P_2} \cdot \mathbf{M}_{P_1}$. As the length of rule increases, it will greatly improve scalability when exploring the entire path of search space grows dramatically.

**Co-occurence Score Function.** Besides synonymy, co-occurrence is also widely studied in natural language processing [Jones *et al.*, 2015]. We can obviously observe a characteristic of the CP rule is that adjacent predicates in the body atoms share the same parametric variables, such as $P_1(x, z)$ sharing variables $z$ with $P_2(z, y)$. It means the rule $r$ is still reasonable when there are large numbers of entities instantiating the parameter $z$. This feature of sharing is called *Co-occurence*.

The co-occurrence score function we use is the same as the definition of **RLvLR**. It is based on the notion of argument embeddings, the average value of the embeddings of all the entities appearing in the position of this argument. Formally, the embeddings of the subject and object argument of a predicate P are defined as:

$$\mathbf{V}_P^s = \frac{1}{n} \sum_{e \in E_P^s} k_e.\mathbf{E} \quad \text{and} \quad \mathbf{V}_P^o = \frac{1}{n} \sum_{e \in E_P^o} l_e.\mathbf{E}$$

where $n = \#\{P(e_s, e_o) \in F'\}$, $E_P^s = \{e_s | \exists e_o$ s.t. $P(e_s, e_o) \in F'\}$, $E_P^o = \{e_o | \exists e_o$ s.t. $P(e_s, e_o) \in F'\}$, $F'$ is defined by sample phase, $k_e = \#\{P(e, e_o) | \exists e_o$ s.t. $P(e, e_o) \in F'\}$ and $l_e = \#\{P(e_s, e) | \exists e_s$ s.t. $P(e_s, e) \in F'\}$. $\mathbf{E}$ denotes the embedding of $e$.

According to the definition of co-occurrence, the embedding of the object argument of $P_1$ should be similar to the embedding of the subject argument of $P_2$, denoted $\mathbf{V}_{P_1}^o \approx \mathbf{V}_{P_2}^s$.

Similarly, in addition to this, there is a shared relationship between $P_t$ and the body atoms: $\mathbf{V}_{P_1}^s \approx \mathbf{V}_{P_t}^s$ and $\mathbf{V}_{P_2}^o \approx \mathbf{P}_{P_t}^o$. Thence, the co-occurrence scoring function is defined as follows:

$$f_{co}(r) = sim(\mathbf{V}_{P_1}^o, \mathbf{V}_{P_2}^s) + sim(\mathbf{V}_{P_1}^s, \mathbf{V}_{P_t}^s)$$
$$+ sim(\mathbf{V}_{P_2}^o, \mathbf{P}_{P_t}^o) \qquad (4)$$

Note that the calculation of $sim$ is similar to that of synonymy.

Finally, after using both of these two scoring functions to complement each other and purne the poor rules, the set of candidates will be saved for the final evaluation.

## 2.4 Evaluate and Filter

The candidate rules that are initially screened by the calculated scoring functions are just a set of results that we obtain by some operations. We also need to use the evaluation function EvaluateAndFilter($\cdot$) to check whether these rules meet the expectations of cognition or the actual situation based on the data. Although KGs are still far from completion, it is necessary to ignore these missings or errors in practice. After being evaluatd by different levels of degrees in $Degree$, the last remaining rule is relatively high quality. Besides, We propose a supervised method of learning the weights of rules, using the TensorLog operation to construct a parametric learning model to achieve the final rule evaluation.

In the previous related work, like [Chen *et al.*, 2016] [Galárraga *et al.*, 2015] and [Omran *et al.*, 2018], standard confidence (SC) and head coverage (HC) usually be used for assessing the quality of rule. [Omran *et al.*, 2018] proposed a efficient method to calculate them in the way of matrix multiplication. In addition to the two degrees, [Galárraga *et al.*, 2015] also proposed an improved metric based on SC called xx, but it has been proven to not achieve good results.

Suppose $r$ is a CP rule in the form of formular 1. The support degree is defined as the number of entity pairs which make the facts, body and head atoms of the rule instantiated with other entities, exist in KG:

$$supp(r) = \#\{(e_0, e_n) | P_{body}(e_0, e_n) \wedge P_t(e_0, e_n)\}$$

where $P_{body}(e_0, e_n)$ means $\exists e_1, \ldots, e_{n-1}$ in the KG such that $P_1(e_0, e_1), P_2(e_1, e_2), \ldots, P_n(e_{n-1}, e_n)$ exist in the KG. We count the number of entity pairs as support instances.

However, only counting the number of occurrences in KG is not able to fully evaluate this rule. Because KG is incomplete, the frequency of different instances varies greatly. Therefore, the support degree should be calculated for satisfying the probability ratio of body and head respectively, which can better indicate the quality.

So the degrees of standard confidence (SC) and head coverage (HC) are defined as follows:

$$SC(r) = \frac{supp(r)}{\#(e_0, e_n) : body(r)(e_0, e_n)} \qquad (5)$$

$$HC(r) = \frac{supp(r)}{\#(e_0, e_n) : P_t(e_0, e_n)} \qquad (6)$$

## 3 Experiments

For the following five benchmarks, we select different datasets for the two tasks of rule mining and link prediction.

Table 1: The specifications of five benchmarks.

| KG | Fact | Entity | Predicate |
|---|---|---|---|
| FB15K-237 | 310116 | 0 | 237 |
| FB75K | 0 | 0 | 13 |
| YAGO2s | 4125967 | 0 | 37 |
| Wikidata | 8397936 | 0 | 430 |
| DBpedia 3.8 | 11024066 | 0 | 650 |

## 3.1 Rule Mining

**Compare with Yang. with FB15k-237.**

**Compare with AMIE+ , Neural LP and RLvLR with DB, Yago2s, Wikidata.**

## 3.2 Link prediction

# 4 Conclusion

## Acknowledgments

## References

[Antoine *et al.*, 2013] Bordes Antoine, Usunier Nicolas, Garcia-Duran Alberto, Weston Jason, and Yakhnenko Oksana. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc., 2013.

[Auer *et al.*, 2007] Soren Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *ISWC*, page 722–735, 2007.

[Barati *et al.*, 2016] Molood Barati, Quan Bai, and Qing Liu. Swarm: An approach for mining semantic association rules from semantic web data. In Richard Booth and Min-Ling Zhang, editors, *PRICAI 2016: Trends in Artificial Intelligence*, pages 30–43, Cham, 2016.

[Chen *et al.*, 2016] Yang Chen, Daisy Zhe Wang, and Sean Goldberg. Scalekb: scalable learning and inference over large knowledge bases. *The VLDB Journal*, 25(6):893–918, Dec 2016.

[Chen *et al.*, 2018] Xuelu Chen, Muhao Chen, Weijia Shi, Yizhou Sun, and Carlo Zaniolo. Embedding uncertain knowledge graphs. *CoRR*, abs/1811.10667, 2018.

[Cohen, 2016] William W. Cohen. Tensorlog: A differentiable deductive database. *CoRR*, abs/1605.06523, 2016.

[Galárraga *et al.*, 2013] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. Amie: Association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13, pages 413–422, New York, NY, USA, 2013. ACM.

[Galárraga *et al.*, 2015] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. Fast rule mining in ontological knowledge bases with amie$$+$$+. *The VLDB Journal*, 24(6):707–730, Dec 2015.

[Jones *et al.*, 2015] Michael N. Jones, Jon Willits, and Simon Dennis. Models of semantic memory. In *Oxford Handbook of Mathematical and Computational Psychology*, page 232–254, 2015.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, and Maosong Sun. Modeling relation paths for representation learning of knowledge bases. *EMNLP*, pages 705–714, 06 2015.

[Mitchell *et al.*, 2018] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, and et al. Never-ending learning. *Communications of the ACM*, 2018.

[Nickel *et al.*, 2012] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: Scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 271–280, New York, NY, USA, 2012. ACM.

[Omran *et al.*, 2018] Pouya Ghiasnezhad Omran, Kewen Wang, and Zhe Wang. Scalable rule learning via learning representation. *Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 2149–2155, July 2018.

[Paritosh *et al.*, 2008] Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250, 2008.

[Speer *et al.*, 2017] R. Speer, Chin J., and Havasi C. Conceptnet 5.5: An open multilingual graph of general knowledge. *AAAI*, 2017.

[Stephen, 1997] Muggleton Stephen. Learning from positive data. In *Inductive Logic Programming.*, pages 358–376. Springer Berlin Heidelberg, 1997.

[Suchanek *et al.*, 2007] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *WWW*, page 697–706, 2007.

[Vrandečić and Krötzsch, 2014] Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledge base. *Communications of the ACM*, 57:78–85, 2014.

[Yang *et al.*, 2015] Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR)*, May 2015.

[Yang *et al.*, 2017] Fan Yang, Zhilin Yang, and William W. Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems 30*, pages 2319–2328. Curran Associates, Inc., 2017.

Table 2: For FB15k-237K, compare the number of rules and time spent on mining with RLVLR.

| Target Predicate | Our | | | | | RLvLR | |
|---|---|---|---|---|---|---|---|
| | Len=3 | Len=4 | Len=5 | Num | Time | Num | Time |
| formOfGovernment | 20 | 239 | 142 | **401** | **1.13h** | 137 | 5h |
| awardWinner | 0 | 0 | 0 | **0** | **h** | 1024 | 5h |
| parentGenre | 0 | 0 | 0 | **0** | **h** | 52 | 5h |
| eventLocation | 18 | 240 | 32 | **290** | **2.56h** | 229 | 5h |