UniKER: A Unified Framework for Combining Embedding and Definite Horn Rule Reasoning for Knowledge Graph Inference

Kewei Cheng[1], Ziqing Yang[2], Ming Zhang[2], and Yizhou Sun[1]
[1]Department of Computer Science, University of California, Los Angeles, Los Angeles, CA
[2]Department of Computer Science, School of EECS, Peking University, Beijing, China
viviancheng@cs.ucla.edu, yangziqing@pku.edu.cn
mzhang_cs@pku.edu.cn, yzsun@cs.ucla.edu

Abstract

Knowledge graph inference has been studied extensively due to its wide applications. It has been addressed by two lines of research, i.e., the more traditional logical rule reasoning and the more recent knowledge graph embedding (KGE). Several attempts have been made to combine KGE and logical rules for better knowledge graph inference. Unfortunately, they either simply treat logical rules as additional constraints into KGE loss or use probabilistic models to approximate the exact logical inference (i.e., MAX-SAT). Even worse, both approaches need to sample ground rules to tackle the scalability issue, as the total number of ground rules is intractable in practice, making them less effective in handling logical rules. In this paper, we propose a novel framework UniKER to address these challenges by restricting logical rules to be definite Horn rules, which can fully exploit the knowledge in logical rules and enable the mutual enhancement of logical rule-based reasoning and KGE in an extremely efficient way. Extensive experiments have demonstrated that our approach is superior to existing state-of-the-art algorithms in terms of both efficiency and effectiveness.

1 Introduction

Knowledge Graphs (KGs) have grown rapidly recently which provide remarkably valuable resources for many real-world applications (Auer et al., 2007; Bollacker et al., 2008; Suchanek et al., 2007). KG reasoning, which aims at inferring missing knowledge through the existing facts, is the key to the success of many downstream tasks and have received wide attention.

Knowledge Graph Embedding (KGE) methods currently hold the state-of-the-art in KG reasoning (Bordes et al., 2013; Wang et al., 2014; Yang et al., 2014; Sun et al., 2019). They aim to capture the similarity of entities via exploring rich structure information in KGs to predict unseen triples. Despite the excellent performance of KGE methods, the ignorance of possible high-order constraints specified by logical rules limits their application in more complex reasoning tasks. In Fig. 1, for example, we can infer *Stilwell* lives in *USA* based on KGE, due to her similarity with *Miller* in terms of embeddings, as both can be reached via the same relation transformation (i.e. *isMarriedTo*) from the same entity (*Edison*). But without the capability of leveraging logical rules, we cannot infer *Stilwell* and *Miller* speak *English*.

An alternative solution is to infer missing facts via logical rules, which have been extensively explored by traditional logical rule-based methods (Richardson and Domingos, 2006; De Raedt and Kersting, 2008). As shown in Fig. 1, two entities (e.g., *Miller* and *English*) that are not directly connected in a KG could participate in the same *ground* logical rule, (e.g., *speakLanguage(Miller, English)* $\leftarrow$ *liveIn(Miller, USA)* $\wedge$ *officialLanguage(USA, English)*), and a relation between them can be inferred if all predicates in the rule body are true. Different from KGE, logical inference treats triples as independent units and ignores the correlation among them. As a result, the performance of logical inference highly depends on the completeness of KGs, which suffers from severe insufficiency in reality. For example, due to the absence of the triple *liveIn(Mary Stilwell, USA)*, the triple *speakLanguage(Mary Stilwell, English)* can not be inferred in Fig. 1. Besides its sensitivity to the quality of KG, logical inference is also known for its high computation complexity as it requires instantiating universally quantified rules into ground rules, which is extremely time-consuming.

Although both embedding-based methods and logical rule-based methods have their limitations, they are complementary for better reasoning capability. As shown in Fig. 1, on one hand, logical rules are useful to provide additional information by exploiting the higher-order dependency of KG relations (Fig. 1(d)). On the other hand, high-

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9753–9771
November 7–11, 2021.

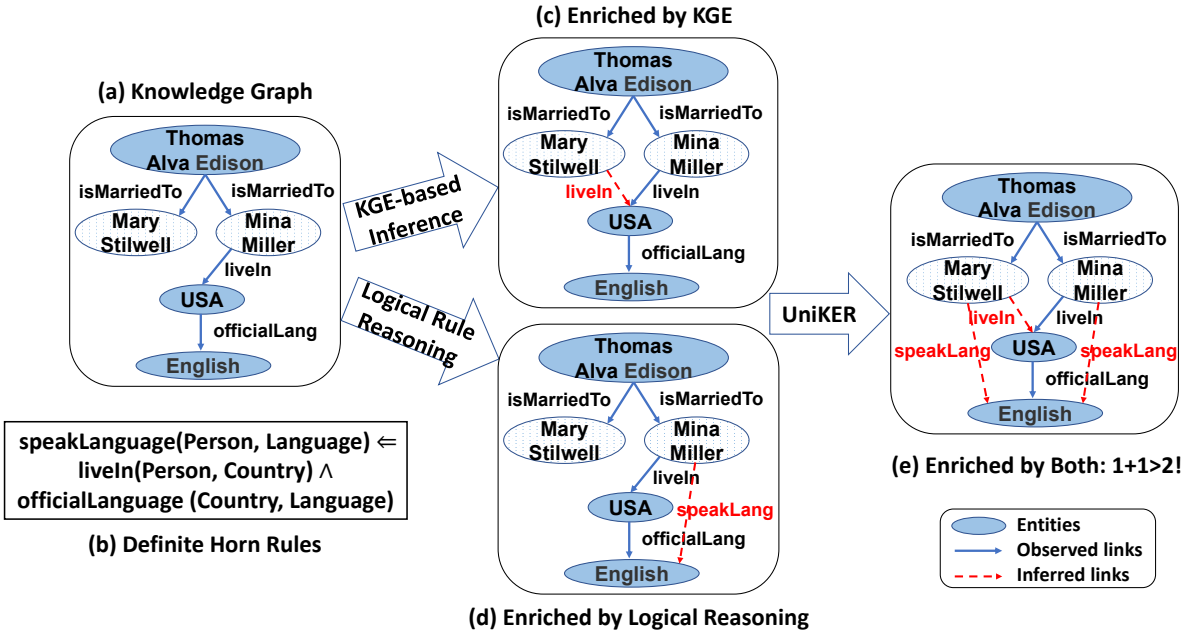

Figure 1: Given (a) a KG with observed facts and (b) a set of definite Horn rules, different inference results can be obtained using (c) KGE, (d) logical inference, and (e) the proposed UniKER approach. The synergy of KGE and logical reasoning via UniKER is more powerful than a simple union of (c) and (d).

quality embedding learned by KGE models, in turn, will help to prepare a more complete KG for logical rule-based reasoning (Fig. 1(c)).

Despite several attempts made to combine KGE and logical rules for KG reasoning, they either simply treat logical rules as additional constraints into KGE loss (Guo et al., 2016; Rocktäschel et al., 2015; Demeester et al., 2016) or use probabilistic model to approximate the exact logical inference (i.e., MAX-SAT) (Qu and Tang, 2019; Zhang et al., 2019; Harsha Vardhan et al., 2020). Moreover, these methods rely on ground rules, the total number of which is intractable in practice. To tackle the scalability issue, only a small portion of ground predicates/ground rules are sampled to approximate the inference process, which causes *further* information loss from the logic side.

To overcome the above issues, we propose a novel **Uni**fied framework for combining **K**nowledge graph **E**mbedding with logical **R**ules (**UniKER**) for better KG reasoning, to handle a special type of first-order logic, i.e., the definite Horn rules. First, we combine logical rule reasoning and KG embedding in an iterative manner, to make sure the inferred knowledge via both techniques can benefit each other as shown in Fig. 1. Second, we propose an iterative grounding algorithm to extend the classic forward chaining algorithm that is designed for definite Horn rule reasoning in an extremely efficient way. Consequently, UniKER can fully exploit the knowledge contained in logical rules and enrich KGs for better embedding. Meanwhile, KGE enhances the forward chaining by including more potential useful hidden facts (See Fig. 1(c)). In this way, two procedures mutually enhance each other. The main contributions of this paper are summarized as follows:

- We investigate the problem of combining embedding and definite Horn rules, a much simpler yet popular form of logical rules, for KG inference.
- A unified framework, UniKER, is proposed, which provides a simple yet effective iterative mechanism to let logical inference and KGE mutually enhance each other in an efficient way.
- We theoretically and experimentally show that UniKER is superior to existing SOTA methods in terms of efficiency and effectiveness.

2 Preliminaries

2.1 Knowledge Graphs in the Language of Symbolic Logic

A KG, denoted by $\mathcal{G} = \{E, R, O\}$, consists of a set of entities E, a set of relations R, and a set of observed facts O. Each fact in O is represented by a triple (e_i, r_k, e_j), where $e_i, e_j \in E$

and $r_k \in R$. In the language of logic, entities can also be considered as **constants** and relations are called **predicates**. Each predicate in KGs is a binary logical function defined over two constants, denoted as $r(\cdot, \cdot)$. **A ground predicate** is a predicate whose arguments are all instantiated by constants. For example, given a predicate *liveIn*$(\cdot, \cdot)$, by assigning constants *Miller* and *USA* to it, we get a ground predicate *liveIn(Miller, USA)*. A triple (e_i, r_k, e_j) is essentially a ground predicate, denoted as $r_k(e_i, e_j)$ in the language of logic. In the reasoning task, a ground predicate can be regarded as a binary random variable: $r_k(e_i, e_j) = 1$ when the triple (e_i, r_k, e_j) holds true, and $r_k(e_i, e_j) = 0$ otherwise. Given the observed facts $\mathbf{v}_O = \{r_k(e_i, e_j) | (e_i, r_k, e_j) \in O\}$, the task of **knowledge graph inference** is to predict the truth value for all **hidden triples** (i.e., unobserved triples) $\mathbf{v}_H = \{r_k(e_i, e_j) | (e_i, r_k, e_j) \in H\}$, where $H = \{(e_i, r_k, e_j) | (e_i, r_k, e_j) \notin O, e_i, e_j \in E, r_k \in R\}$.

2.2 First Order Logic and Definite Horn Rules

First-order logic (FOL) rules are constructed over predicates using logical connectives and quantifiers, which usually require extensive human supervision to create and validate and thus severely limit their applications. Instead, **definite Horn rules**, as a special case of FOL rules, can be extracted automatically and efficiently via modern rule mining systems, such as WARMR (Dehaspe and Toivonen, 1999) and AMIE (Galárraga et al., 2015) with high quality, which are widely used in practice. Definite Horn rules are composed of a body of conjunctive predicates and a single positive head predicate. They are usually written in the form of implication as shown below:

$$\forall x, y \, r_0(x, y) \leftarrow r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge r_3(z_2, y) \quad (1)$$

where $r_0(x, y)$ is called the **head** of the rule while $r_1(x, z_1) \wedge r_2(z_1, z_2) \wedge r_3(z_2, y)$ is the **body** of the rule. By substituting the variables x, z_1, z_2, y with concrete entities e_i, e_p, e_q, e_j, we get a ground definite clause as follows:

$$r_0(e_i, e_j) \leftarrow r_1(e_i, e_p) \wedge r_2(e_p, e_q) \wedge r_3(e_q, e_j) \quad (2)$$

2.3 Logical Reasoning

Traditional logical inference aims to find an assignment of truth values to all hidden ground predicates, leading to maximizing satisfied ground rules. Thus, it can be mathematically modeled as a MAX-SAT problem, which is NP-hard (Shimony, 1994).

2.4 Knowledge Graph Embedding

KGE aims to capture the similarity of entities by embedding entities and relations into low-dimensional vectors. Scoring functions, which measure the plausibility of triples in KGs, are the crux of KGE models. We denote the score of a triple (e_i, r_k, e_j) calculated following scoring function as $f_{r_k}(e_i, e_j)$. Representative KGE algorithms include TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransR (Lin et al., 2015), DistMult (Yang et al., 2014), ComplEx (Trouillon et al., 2016), and RotatE (Sun et al., 2019), which differ in their scoring functions.

3 Related Work on Integrating Embedding and Logical Rules

MAX-SAT problem is defined using boolean logic while scoring function of KGE provides a soft truth value to triples in KG. Probabilistic logic is widely used to integrate both worlds into the same framework, which is able to extend boolean logic to probabilistic logic to enable uncertain inference. These approaches can be divided into two categories: (1) designing a Probablistic Soft Logic (PSL)-based regularization to embedding models and (2) designing an embedding-based variational distribution for variational inference of Markov Logic Network (MLN) (Richardson and Domingos, 2006), a probabilistic logic that applies Markov network to FOL.

PSL-based Regularization in Embedding Loss. The first way to combine two worlds is to treat logical rules as additional regularization to embedding models, where the satisfaction loss of ground rules is integrated into the original embedding loss. Probabilistic Soft Logic (PSL) (Bach et al., 2015) is used to compute the satisfaction loss, where the probability of each predicate is determined by the embedding. KALE (Guo et al., 2016), RUGE (Guo et al., 2017) and Rocktäschel et al. (Rocktäschel et al., 2015) are some of the representative methods. A summary of these methods can be found in Appendix E. All approaches in this category have to instantiate universally quantified rules into ground rules before model learning. When including all ground rules into the calculation of satisfaction loss, the additional regularization becomes the convex program which reasons analogous MAX-SAT problem defined over Lukasiewicz logic (Klir and Yuan, 1996) whose solution is an approximation to MAX-SAT problem (Bach et al., 2015). Detailed proof

is given in Appendix C. As the total number of ground rules is intractable in practice, only a small portion of ground rules will be sampled to tackle the scalability issue, which further leads to the loss of logical information. Moreover, most methods in this category make only a one-time injection of logical rules to enhance embedding, ignoring the interactive nature between embedding and logical inference (Guo et al., 2016; Rocktäschel et al., 2015).

Embedding-based Variational Inference for MLN. The second type extends Markov Logic Network (MLN) (Richardson and Domingos, 2006) instead. Several methods including pGAT (Harsha Vardhan et al., 2020), ExpressGNN (Zhang et al., 2019) and pLogicNet (Qu and Tang, 2019) are proposed to leverage graph embedding to define variational distribution for all possible hidden triples to conduct variational inference of MLN. A detailed introduction of these methods can be found in Appendix F. One main drawback is that they only approximate the exact logical inference (i.e., MAX-SAT) due to the nature of the approximate solution provided by variational inference. Besides, inference efficiency is another challenge. Given the fact that KGs usually contain a large number of entities, it is impractical to optimize on all hidden triples. Therefore, only a small portion of hidden triples are sampled to reduce the computational complexity. This brings in the similar issue of information loss from the logical side.

4 A Unified Framework for Knowledge Graph Inference: UniKER

Rather than follow probabilistic logic to integrate logical rules and KGE, we show that by leveraging the nice properties of definite Horn rules, there is a much simpler way to directly derive an optimal boolean solution for MAX-SAT problem. To capture the mutual interaction between KGE and logical inference, we proposed an iterative mechanism, which ensures that UniKER is more potent than a simple union of KGE and logical rules.

4.1 Update KG via Forward Chaining-based Logical Reasoning

We prove that there exists a truth assignment that satisfies all ground rules when restricting logical rules to be definite Horn rules. Detailed proof can be found in Appendix D. The next question is how to conduct such an assignment efficiently. We denote the satisfying truth assignment as $\mathbf{v}_H^{T*}$ (hidden triples that are true) and $\mathbf{v}_H^{F*}$ (hidden triples that are false), i.e., $\mathbf{v}_H^{T*} = \{r_k(e_i, e_j) = 1 \mid r_k(e_i, e_j) \in \mathbf{v}_H\}$ and $\mathbf{v}_H^{F*} = \{r_k(e_i, e_j) = 0 \mid r_k(e_i, e_j) \in \mathbf{v}_H\}$. An existing algorithm called **forward chaining** (Salvat and Mugnier, 1996) can derive $\mathbf{v}_H^{T*}$ and $\mathbf{v}_H^{F*}$ efficiently. Starting from known facts (e.g., *liveIn (Mina Miller, USA)*, *officialLanguage (USA, English)*), it triggers all ground rules whose premises are satisfied (e.g., *speakLanguage (Mina Miller, English)* $\leftarrow$ *liveIn (Mina Miller, USA)* $\wedge$ *officialLanguage (USA, English)*), and adds their conclusion (e.g., *speakLanguage (Mina Miller, English)*) to the known facts until no facts can be added anymore. As illustrated in Fig. 1, unlike other logical inference algorithms, which require all ground predicates (including both observed and unobserved ground predicates) into calculation, forward chaining adopts "lazy inference" instead. It involves only a small subset of "active" ground predicates/rules, and activates more if necessary as the inference proceeds. The mechanism dramatically improves inference efficiency by avoiding the computation for massive ground predicates/rules that are never used. Moreover, considering that definite Horn rules which can be extracted efficiently via modern rule mining systems are usually chain-like Horn rules, which is in the form as shown in Eq. (1). The conjunctive body of a ground chain-like Horn rules is essentially a path in a KG, which can be extracted efficiently using sparse matrix multiplication. More general implementation of forward chaining algorithm can be found in Appendix G.

4.2 Update KG via Embedding-based Inference

KG Embedding on the Updated KG. Since $\mathbf{v}_H^{T*}$ and $\mathbf{v}_H^{F*}$ are the satisfying truth assignment derived by forward chaining, knowledge contained in definite Horn rules is guaranteed to be fully exploited. Thus, we can treat (1) both observed triples O and the newly inferred triples $\mathbf{v}_H^{T*}$ as positive triples and (2) $\mathbf{v}_H^{F*}$ as negative triples to form the objective function of KGE model:

$$\mathcal{L} = \sum_{(e_i, r_k, e_j) \in \{O \cup \mathbf{v}_H^{T*}\}} \mathcal{L}(e_i, r_k, e_j) \tag{3}$$

in which $\mathcal{L}(e_i, r_k, e_j)$ is defined as:

$$\max(0, \gamma - f_{r_k}(e_i, e_j) + \sum_{(e_i', r, e_j') \in \mathcal{N}(e_i, r, e_j)} f_{r_k}(e_i', e_j') \tag{4}$$

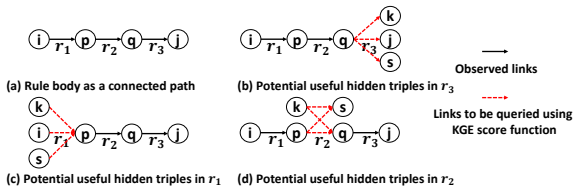

Figure 2: Illustration of potential useful hidden triples by taking Eq. (2) as an example.

where (e_i', r_k, e_j') denotes their corresponding negative samples, and γ is a margin to separate them. The score $f_{r_k}(e_i, e_j)$ of a triple (e_i, r_k, e_j) can be calculated following any scoring functions of KGE models. To reduce the effects of randomness, we sample multiple negative triples for each positive sample, which is denoted as $\mathcal{N}(e_i, r_k, e_j)$. To ensure true but unseen triples will not be sampled, the selection of $\mathcal{N}(e_i, r_k, e_j)$ is restricted to $\mathbf{v}_H^{F*}$.

Update KG with KGE-based Inference. Although forward chaining can find the satisfying truth assignment for all hidden triples efficiently, its reasoning ability is severely limited by the coverage of rules, the incompleteness of KGs, and the errors/noise contained in KGs. Considering its strong reasoning ability and robustness, KGE models are not only useful to (1) prepare a more complete KG by adding useful hidden triples but also helpful to (2) eliminate incorrect triples in both KGs and inferred results.

(1) Including Potential Useful Hidden Triples ($\Delta+$). Since the body of a definite Horn rule is a conjunction of predicates, its ground rule can get activated and contribute to logical inference only if all the predicates in its body are completely observed. Due to the sparsity of real-world KGs, only a small portion of ground rules can participate in logical inference, which severely limits the reasoning ability of definite Horn rules. A straightforward solution would be computing the score for every hidden triple and adding the most promising ones with the highest scores to the KG. Unfortunately, the number of hidden triples is quadratic to the number of entities (i.e. $O(|R||E|^2)$), thus it is too expensive to compute scores for all of them. Instead, we adopt "lazy inference" strategy to select only a small subset of "potentially useful" triples. Take the ground rule in Eq. (2) as an example, if $r_1(e_i, e_p) \in v_O$, $r_3(e_q, e_j) \in v_O$, and $r_2(e_p, e_q) \in v_H$, we would not be able to infer the head $r_0(e_i, e_j)$ as whether $r_2(e_p, e_q)$ is true or not is unknown. Thus, $r_2(e_p, e_q)$ becomes the crux to determine the truth value of the head, which is called "potentially useful". In general, given a ground rule whose body includes only one unobserved ground predicate, this unobserved ground predicate can be regarded as a "potentially useful" triple. We denote the set of all "potentially useful" triples as Δ_+. According to their positions, "potentially useful" triples can be divided into two categories: (1) triples that are the first or the last predicate in a ground rule; and (2) triples that are neither the first nor the last. We propose algorithms to identify both types of "potentially useful" triples respectively as illustrated in Fig. 2 by taking Eq. (2) as an example. More details are summarized in Appendix A. Score $f_{r_k}(e_i, e_j)$ will be computed by KGE model to predict whether a "potentially useful" triple is true. If $f_{r_k}(e_i, e_j)$ is larger than the given threshold Ψ, the triple is classified as true. Otherwise, the triple is classified as false. And we experimentally analysed the effect of Ψ in the Appendix J.2. Note that a dynamic programming algorithm can also be used to alleviate the computational complexity for long rules. The detailed algorithm can be found in the Appendix B.

(2) Excluding Potential Incorrect Triples ($\Delta-$). In addition, due to the symbolic nature, logical rules cannot handle noisy data as well. If the KGs contain any error, based on incorrect observations, forward chaining will not be able to make the correct inference. Even worse, it may contribute to the propagation of the error by including incorrectly inferred triples into KGs. Therefore, a clean KG is significant for logical inference. Since KGE models show great power in capturing the network structure of KGs, incorrect triples usually result in contradictions and get lower prediction scores in KGE models compared to correct ones. Therefore, score $f_{r_k}(e_i, e_j)$ computed by KGE model is able to measure reliability of triple (e_i, r_k, e_j) in $O \cup \mathbf{V}_H^{T*}$. We denote bottom $\theta\%$ triples with lowest prediction scores as Δ_-. It will be excluded from $O \cup \mathbf{V}_H^{T*}$ to alleviate the impact of noise.

4.3 Integrating Embedding and Logical Rules in an Iterative Manner

Since logical rules and KGE can mutually enhance each other, we propose a unified framework, known as UniKER, to integrate KGE and definite Horn rules-based inference iteratively. The pseudo-code of UniKER can be found in Algorithm 1. MAX_ITER is the user specified max

Algorithm 1: Learning Procedure of UniKER

Input: Observed facts in knowledge bases O; threshold to eliminate noise $\theta\%$; threshold to include useful hidden triples Ψ; a set of definite Horn rules $\mathcal{F}$

Output: KG embeddings

for $t = 1$: *MAX_ITER* **do**
  // Update KG via Logical Reasoning
  Derive $\mathbf{v}_H^{T\,t*}$ from O and update $O \leftarrow O \cup \mathbf{v}_H^{T\,t*}$
  // Update KG via Embedding-based Inference
  KG embedding learning based on O;
  Compute Δ_- and update $O \leftarrow O - \Delta_-$;
  Compute Δ_+ and update $O \leftarrow O \cup \Delta_+$;
end

iterations to run the algorithm, which highly depends on KG datasets. According to results in Fig. 4, MAX_ITER is usually set as 2 to 4. For each iteration of UniKER, it is comprised of two steps. First, we focus on logical reasoning to update KG. Following forward chaining algorithm, by triggering all rules whose premises are satisfied, we derive entailed triple set $\mathbf{v}_H^{T\,t*}$ at t-th iteration, which is a subset of $\mathbf{v}_H^{T\,*}$ ($\mathbf{v}_H^{T\,*} = \cup_{t=1}^{+\infty} \mathbf{v}_H^{T\,t*}$). Then, the newly inferred triples $\mathbf{v}_H^{T\,t*}$ are added to KG by updating $O = O \cup \mathbf{v}_H^{T\,t*}$. Second, we focus on embedding-based inference to update KG. KGE can be learned based on the updated KG after the first step. With the learned embeddings, Δ_-, which is the bottom $\theta\%$ triples with lowest prediction scores, can be eliminated from O. Meanwhile, Δ_+, which are potentially useful hidden triples, can be added to O.

4.4 Connection to Existing Approaches

Connection to PSL-based Regularization Approaches. The general objective of PSL-based regularization approaches can be written as:

$$\mathcal{L}_{KGE} + \lambda \mathcal{L}_{PSL} \tag{5}$$

where $\mathcal{L}_{KGE}$ denotes the loss of the base KGE model while $\mathcal{L}_{PSL}$ corresponds to the satisfaction loss of the sampled ground rules. When including all ground rules into the calculation of $\mathcal{L}_{PSL}$, $\mathcal{L}_{PSL}$ becomes the convex program which reasons analogous MAX-SAT problem defined over Lukasiewicz logic, which only approximates the exact logical inference. Detailed proof is given in Appendix C. Instead of guiding the embedding learning approximately, UniKER directly take the optimum of MAX-SAT problem as targets to optimize the embedding model. Thus it can better exploit the knowledge contained in definite Horn rules. Moreover, $\mathcal{L}_{PSL}$ makes only a one-time injection of logical rules to enhance embedding, where logical reasoning will not be further enhanced even after the KGE gets improved. On the contrary, UniKER is able to capture the interactive nature between embedding and logical inference.

Connection to Embedding-based Variational Inference to MLN. The general objective of embedding-based variational inference for MLN can be written as:

$$\mathcal{L}_{ELBO}(Q_\theta, P_w) + \lambda \mathcal{L}_{KGE}(Q_\theta) \tag{6}$$

where the variational distribution Q_θ is defined using a KGE model and P_w is the true posterior defined over MLN. $\mathcal{L}_{KGE}(Q_\theta)$ denotes the loss of the base KGE model. By optimizing $\mathcal{L}_{ELBO}(Q_\theta, P_w)$, the KL divergence between Q_θ and P_w can be minimized. In this way, the knowledge contained in rules can be transferred into the embeddings. Due to the nature of the approximate solution provided by variational inference and the information loss caused the sampling procedure, Q_θ can only approximate the optimum of MAX-SAT problem and no guarantees are provided on the quality of the solutions obtained. Instead of guiding the learning of embedding model via variational inference, we directly solve MAX-SAT problem and use the derived knowledge $\mathbf{v}_H^{T\,*}$ to train the embedding model, which leads to superior reasoning.

Advantages of UniKER compared to SOTA methods. We categorize all existing methods according to two aspects: (1) whether they capture mutual interaction between KGE and logical inference; and (2) whether they conduct exact logical inference. The summary is given in Table 1. For the first aspect, most PSL-based regularization approaches make only a one-time injection of logical rules to enhance embedding, while embedding-based variational inference to MLN and UniKER provide the interaction between embedding and logical inference. For the second aspect, both PSL-based regularization approaches and embedding-based variational inference to MLN follow the framework of probabilistic logic to combine logical rule and KGE, which can only approximate the optimal solution of MAX-SAT problem. UniKER is the first to use forward chaining to conduct exact inference, which provides an optimal solution to the original MAX-SAT problem.

Categories	Methods	Interactive	Exact Logical Inference
PSL-based Regularization	KALE (Guo et al., 2016)	×	×
	RUGE (Guo et al., 2017)	✓	×
	Rocktäschel et al. (Rocktäschel et al., 2015)	×	×
Embedding-based Variational Inference to MLN	pLogicNet (Qu and Tang, 2019)	✓	×
	ExpressGNN (Zhang et al., 2019)	✓	×
	pGAT (Harsha Vardhan et al., 2020)	✓	×
Our Proposed Method	UniKER	✓	✓

Table 1: Capabilities of different methods.

5 Experiments

5.1 Experimental Setup

Datasets. We implement experiments on three large-scale real-world KGs (i.e., Family (Denham, 1973), FB15k-237 (Bordes et al., 2013) and WN18RR (Bordes et al., 2013)). AMIE+ (Galárraga et al., 2015) is used to generate candidate rules automatically. More details about datasets, rule generation and examples of logical rules are provided in Appendix I and J.5.

Compared Methods. We evaluate our proposed method against SOTA algorithms, including (1) basic KGE models (e.g., RESCAL (Nickel et al., 2011), SimplE (Kazemi and Poole, 2018), HyPER (Balažević et al., 2019a), TuckER (Balažević et al., 2019b), TransE (Bordes et al., 2013), DistMult (Yang et al., 2014) and RotatE (Sun et al., 2019)), (2) traditional logical rule-based methods (e.g., MLN (Richardson and Domingos, 2006) and BLP (De Raedt and Kersting, 2008)), (3) two classes of approaches combining embedding model with logical rules (two representative methods KALE (Guo et al., 2016) and RUGE (Guo et al., 2017) for PSL-based regularization approaches, and pLogicNet (Qu and Tang, 2019), ExpressGNN (Zhang et al., 2019) and pGAT (Harsha Vardhan et al., 2020) for embedding-based variational inference of MLN), and (4) other approaches to combining embedding model with logical rules (e.g., BoxE (Abboud et al., 2020)). To show that UniKER can be easily generalized to various KGE models, we chose TransE (Bordes et al., 2013), DistMult (Yang et al., 2014) and RotatE (Sun et al., 2019) as the scoring function for UniKER.

5.2 KG Completion

To compare different algorithms on KG inference task, we mask the head or tail entity of each test triple, and require each method to predict the masked entity. More detailed settings are in Appendix J.1. Table 2 shows the comparison results, from which we find that: (1) UniKER consistently outperforms KGE models in most cases with significant performance gain, which can ascribe to the utilization of additional knowledge from logical rules; (2) UniKER also obtains better performance than both classes of approaches which combine embedding model with logical rules as it provides an exact optimal solution to satisfiable problem defined over all ground rules rather than employ sampling strategies to do approximation.

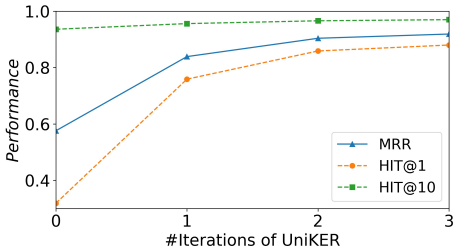

Figure 3: Impact of #iterations on UniKER (KG completion task on Family dataset).

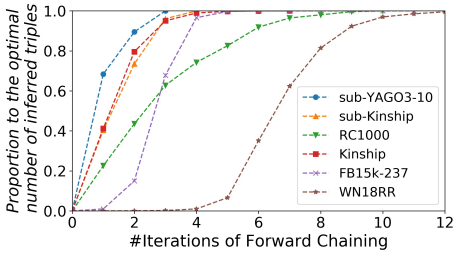

Figure 4: Proportion to the optimal number of inferred triples w.r.t. #iterations for efficiency analysis of Forward Chaining.

Impact of Iterative Algorithm on KG Completion. To investigate how iterative process helps improve reasoning ability of UniKER, we conduct experiments on Family dataset and record the performance of UniKER on test data in terms of Hit@1, Hit@10 and MRR in every iteration. In

Model	Family			FB15k-237			WN18RR		
	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR
RESCAL	0.489	0.894	0.639	0.108	0.322	0.179	0.123	0.239	0.162
SimplE	0.335	0.888	0.528	0.150	0.443	0.249	0.290	0.351	0.311
HypER†	0.364	0.903	0.551	0.252	0.520	0.341	0.436	0.522	0.465
TuckER†	0.373	0.898	0.567	0.266	0.544	0.358	**0.443**	0.526	0.470
BLP†	-	-	-	0.062	0.150	0.092	0.187	0.358	0.254
MLN	0.655	0.732	0.694	0.067	0.160	0.098	0.191	0.361	0.259
KALE	0.433	0.869	0.598	0.131	0.424	0.230	0.032	0.353	0.172
RUGE	0.495	0.962	0.677	0.098	0.376	0.191	0.251	0.327	0.280
ExpressGNN	0.105	0.282	0.164	0.150	0.317	0.207	0.036	0.093	0.054
pLogicNet	0.683	0.874	0.768	0.261	0.567	0.364	0.301	0.410	0.340
pGAT†	-	-	-	0.377	0.609	0.457	0.395	0.578	0.459
BoxE†	-	-	-	-	0.538	0.337	-	0.541	0.451
TransE	0.221	0.874	0.453	0.231	0.527	0.330	0.007	0.406	0.165
UniKER-TransE	0.873	**0.971**	0.916	0.463	**0.630**	0.522	0.040	0.561	0.307
DistMult	0.360	0.885	0.543	0.220	0.486	0.308	0.304	0.409	0.338
UniKER-DistMult	0.770	0.945	0.823	**0.507**	0.587	0.533	0.432	0.538	0.485
RotatE	0.787	0.933	0.862	0.237	0.526	0.334	0.421	0.563	0.469
UniKER-RotatE	**0.886**	**0.971**	**0.924**	0.495	0.612	**0.539**	0.437	**0.580**	**0.492**

† Results on FB15k-237 and WN18RR are taken from the original papers.

Table 2: Effectiveness on KG completion task

Model	θ	Hit@1	Hit@10	MRR
TransE	-	0.026	0.800	0.319
UniKER-TransE	10	0.286	0.776	0.466
	20	0.311	0.816	0.503
	30	0.322	**0.833**	0.520
	40	**0.352**	0.812	**0.523**
	50	0.292	0.791	0.486

Table 3: Ablation study on noise threshold $\theta\%$ on Family dataset (whose train set is injected with noise)

Model	ψ	Hit@1	Hit@10	MRR
UniKER-TransE	10	0.873	0.971	0.916
UniKER-TransE	20	**0.878**	0.972	**0.919**
UniKER-TransE	30	0.873	0.972	0.916
UniKER-TransE	40	0.874	**0.973**	0.917
UniKER-TransE	50	0.871	0.970	0.915

Table 4: Results of Reasoning on Family Dataset with Different Thresholds ($\psi\%$) to Include Useful Hidden Triples.

particular, KGE model is trained based on the original data without any inferred triples included in iteration 0. As presented in Fig. 3, we observed that (1) with the increase of iterations, the performance improves rapidly first, and gradually slows down; (2) UniKER has a bigger impact on Hit@k compared to MRR.

Robustness Analysis. To investigate the robustness of UniKER, we compare the reasoning ability of UniKER with TransE on Family dataset with noise. Complete details of injecting noise are summarized in Appendix J.1. We vary θ among $\{10, 20, 30, 40, 50\}$ to study the effect of the threshold used to eliminate noisy triples. The comparison results are presented in Table 3. We observe that (1) UniKER outperforms TransE on noisy KG with significant performance gain; (2) with the increase of θ, the performance first increases and then decreases. The best performance is achieved when $\theta = 40\%$.

Effect of Threshold Ψ Used to Include Potential Useful Hidden Triples To investigate effect of threshold used to include useful hidden triples, we also compare the reasoning ability of UniKER with TransE on Family dataset with different thresholds Ψ. As threshold can vary a lot for different data sets, to propose a unified way to determine proper threshold Ψ, we take score $f_{r_k}(e_i, e_j)$ corresponding to the triple which ranks as top $\psi\%$ in test dataset as threshold Ψ. We vary ψ among $\{10, 20, 30, 40, 50\}$. The comparison results are presented in Table 9. We can observe that reasoning ability of UniKER does not vary a lot with different thresholds. In other words, the performance is less sensitive to the parameter ψ, which is appealing in practice.

5.3 Efficiency Analysis

Besides the promising results on KG reasoning, our UniKER is superior in terms of efficiency. Though we have theoretically analyzed the computational complexity of UniKER in Appendix H, the efficiency of forward chaining highly depends on KG datasets. Note that forward chaining learns the

optimal truth assignment for the satisfiable problem iteratively, the number of iterations required to achieve the optimal solution may influence its scalability. We first conduct two experiments on six datasets (details are introduced in Appendix I): (1) as presented in Fig. 4, we record the proportion of inferred triples accumulated in every iteration over all inferred triples. The result shows that forward chaining can achieve the optimal solution within 12 iterations, and infer most correct triples within only 4 iterations; (2) as illustrated in Appendix J.3, we evaluate the scalability of forward chaining against a number of SOTA inference algorithms for MLN (e.g., MCMC (Carlo, 2004), MC-SAT (Poon and Domingos, 2006), BP (Yedidia et al., 2001), liftedBP (Singla and Domingos, 2008) and Tuffy (Niu et al., 2011)). Forward chaining runs $100 - 100,000$ times faster than them. Some widely used algorithms MCMC and MC-SAT even cannot handle RC1000 dataset, which indicates the scalability of UniKER. Then, we compared the overall efficiency of our proposed UniKER with other methods. As shown in Appendix J.3, the time cost per epoch of UniKER is shorter than other methods combining embedding with logical rules experimentally, indicating its efficiency.

5.4 Mutual Enhancement between KGE and Logical Inference

In this section, we aim to show that the synergy of KGE and logical inference via UniKER is more powerful than a plain union.

Enhancement of Logical Inference via KGE. On one hand, high quality embedding learned by KGE models is useful to prepare more complete KGs via including useful hidden triples, which the performance of logical inference highly depends on. To show the benefit brought by KGE over logical inference, we evaluate UniKER-TransE against forward chaining on Family Dataset with the triple classification task, which aims to predict correct facts in the testing data. In order to create a testing set for classification, we randomly corrupt relations of correct testing triplets for negative triples construction. It results in a total of $2 \times \#$Test triplets with equal number of positive and negative examples. During evaluation, we adopt three evaluation metrics, i.e., precision, recall and F1. As shown in Table 5, we can observe that although the precision slightly decreases, UniKER outperforms forward chaining with significant performance gain in terms of recall and F1, which validates the enhancement brought by KGE model over logical inference.

Model	Precision	Recall	F1
Forward Chaining	**1.000**	0.919	0.958
UniKER-TransE	0.991	**0.955**	**0.973**

Table 5: UniKER-TransE v. Forward Chaining on Family dataset (whose test set only retains triples that can be inferred by logical rules) on triple True/False classification task.

Model	Hit@1	Hit@3	Hit@10	MRR
TransE	0.267	0.651	0.803	0.476
UniKER-TransE	**0.710**	**0.866**	**0.904**	**0.816**

Table 6: Results of reasoning of UniKER-TransE v.s. TransE on Family dataset (whose test set eliminates triples that can derive from logical rules).

Enhancement of KGE via Logical Inference. On the other hand, logical rules are useful to gather more reliable triples for KGE by exploiting symbolic compositionality of KG relations, which leads to the enhancement of the reasoning ability of KGE model. To investigate the added value brought by logical rules over KGE, we evaluate UniKER-TransE against TransE on Family dataset on the KG completion task. As some triples in test dataset can be directly derived from logical rules, to ensure the improvement comes from the reasoning ability enhancement of KGE model, we exclude the triples derived directly from rules from the test data. As presented in Table 6, we can observe that UniKER-TransE outperforms TransE model with huge performance gain, especially in terms of Hit@1, which can ascribe to the added value brought by logical rules over KGE.

6 Conclusion

In this paper, we proposed a novel framework, known as UniKER, to integrate embedding and definite Horn rules in an iterative manner for better KG inference. We have shown that UniKER can fully leverage the knowledge in definite Horn rules and completely transfer them into the embeddings in an extremely efficient way.

Acknowledgements

This work was partially supported by NSF III-1705169, NSF 1937599, DARPA HR00112090027, Okawa Foundation Grant, and Amazon Research Awards.

References

Ralph Abboud, Ismail Ilkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. *arXiv preprint arXiv:2007.06267*.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. 2015. Hinge-loss markov random fields and probabilistic soft logic. *arXiv preprint arXiv:1505.04406*.

Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019a. Hypernetwork knowledge graph embeddings. In *International Conference on Artificial Neural Networks*, pages 553–565. Springer.

Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019b. Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Chain Monte Carlo. 2004. Markov chain monte carlo and gibbs sampling. *Lecture notes for EEB*, 581.

Luc De Raedt and Kristian Kersting. 2008. Probabilistic inductive logic programming. In *Probabilistic Inductive Logic Programming*, pages 1–27. Springer.

Luc Dehaspe and Hannu Toivonen. 1999. Discovery of frequent datalog patterns. *Data Mining and knowledge discovery*, 3(1):7–36.

Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2016. Lifted rule injection for relation embeddings. *arXiv preprint arXiv:1606.08359*.

Woodrow W Denham. 1973. *The detection of patterns in Alyawara nonverbal behavior*. Ph.D. thesis, University of Washington, Seattle.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. 2015. Fast rule mining in ontological knowledge bases with amie+. *The VLDB Journal—The International Journal on Very Large Data Bases*, 24(6):707–730.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly embedding knowledge graphs and logical rules. pages 192–202.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2017. Knowledge graph embedding with iterative guidance from soft rules.

L Vivek Harsha Vardhan, Guo Jia, and Stanley Kok. 2020. Probabilistic logic graph attention networks for reasoning. In *Companion Proceedings of the Web Conference 2020*, WWW '20, page 669–673, New York, NY, USA. Association for Computing Machinery.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems*, pages 4284–4295.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

George J Klir and Bo Yuan. 1996. Fuzzy sets and fuzzy logic: theory and applications. *Possibility Theory versus Probab. Theory*, 32(2):207–208.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

Radford M Neal and Geoffrey E Hinton. 1998. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *ICML*, volume 11, pages 809–816.

Feng Niu, Christopher Ré, AnHai Doan, and Jude Shavlik. 2011. Tuffy: Scaling up statistical inference in markov logic networks using an rdbms. *Proceedings of the VLDB Endowment*, 4(6):373–384.

Hoifung Poon and Pedro Domingos. 2006. Sound and efficient inference with probabilistic and deterministic dependencies. In *AAAI*, volume 6, pages 458–463.

Meng Qu and Jian Tang. 2019. Probabilistic logic neural networks for reasoning.

Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine learning*, 62(1-2):107–136.

Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. 2015. Injecting logical background knowledge into embeddings for relation extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129.

Eric Salvat and Marie-Laure Mugnier. 1996. Sound and complete forward and backward chainings of graph rules. In *International Conference on Conceptual Structures*, pages 248–262. Springer.

Solomon Eyal Shimony. 1994. Finding maps for belief networks is np-hard. *Artificial intelligence*, 68(2):399–410.

Parag Singla and Pedro M Domingos. 2008. Lifted first-order belief propagation. In *AAAI*, volume 8, pages 1094–1099.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence*.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Jonathan S Yedidia, William T Freeman, and Yair Weiss. 2001. Generalized belief propagation. In *Advances in neural information processing systems*, pages 689–695.

Yuyu Zhang, Xinshi Chen, Yuan Yang, Arun Ramamurthy, Bo Li, Yuan Qi, and Le Song. 2019. Can graph neural networks help logic reasoning?

A Illustration of Potential Useful Hidden Triples

Let every relation r_k in KG associate with an $|E| \times |E|$ matrix $\mathbf{M}^{(k)}$, in which the element $\mathbf{M}^{(k)}_{ij} = 1$ if the triple $(e_i, r_k, e_j) \in O$, and 0 otherwise. The algorithms to identify both types of "potential useful" triples as illustrated in Fig. 2 in main context are given as follows.

- When the "potential useful" triple is the first or the last predicate in a ground rule, other observed triples in a chain-like definite Horn rule still constitute a complete path, which can be extracted efficiently by sparse matrix multiplication. Take Fig. 2 (c) in main context as an example, to identify the "potential useful" triple $r_1(e_i, e_p)$, we have to first extract all connected path $r_2(e_p, e_q) \wedge r_3(e_q, e_j)$ by calculating $\mathbf{M} = \mathbf{M}^{(2)}\mathbf{M}^{(3)}$, where $\mathbf{M}^{(2)}$ and $\mathbf{M}^{(3)}$ are adjacency matrices corresponding to relations r_2 and r_3. Each nonzero element $\mathbf{M}_{pj}$ indicates a connected path between e_p and e_j. We denote all indexes correspond to nonzero rows in $\mathbf{M}$ as $\delta = \{p|(\sum_j \mathbf{M}_{pj}) \neq 0\}$, which indicates that there is always a connected path starting at p. For specific $p \in \delta$, $\Delta_p = \{(e_i, r_1, e_p)|e_i \in E\}$ defines a set "potential useful" triples. If (e_i, r_1, e_p) in Δ_p is predicted to be true via KGE, the head predicates $r_0(e_i, e_j)$ can be inferred.

- Otherwise, the path corresponds to the conjunctive body of the ground rule get broken into two paths by the "potential useful" triple, which we have to extract separately. As shown in Fig.2 (d) in main context , when identifying "potential useful" triples $r_2(e_p, e_q) \in v_H$, two paths to be extracted are essentially two single relations r_1 and r_3, whose corresponding matrices are $\mathbf{M}^{(1)}$ and $\mathbf{M}^{(3)}$, respectively. We denote all indexes correspond to nonzero columns in $\mathbf{M}^{(1)}$ as $\delta_1 = \{p|(\sum_i \mathbf{M}^{(1)}_{ip}) \neq 0\}$ and all indexes correspond to nonzero rows in $\mathbf{M}^{(3)}$ as $\delta_2 = \{q|(\sum_j \mathbf{M}^{(3)}_{qj}) \neq 0\}$. $\Delta_{12} = \{(e_p, r_2, e_q)|p \in \delta_1, q \in \delta_2\}$ defines a set "potential useful" triples. If (e_p, r_2, e_q) in Δ_{12} is predicted to be true via KGE, the head predicates $\{r_0(e_i, e_j)|\mathbf{M}^{(1)}_{ip} \neq 0, \mathbf{M}^{(3)}_{qj} \neq 0\}$ can be inferred.

B Dynamic Programming to Calculate $\mathbf{M}$ for Long Rules

Considering that the body of a chain-like definite Horn rules F_k can be regard as sequence of relations, which can be denoted as $B(F_k) = [r_n, \ldots, r_m]$. As mentioned in subsection Including Potential Useful Hidden Triples, to identify all "potential useful" triples for rule F_k, we have to compute all possible sub-sequences of its body. To alleviate computational complexity, we adopt a dynamic programming algorithm. In particular, we maintain a table T to record all previous calculation and compute new results based on T. The keys of T are all possible sub-sequences of $B(F_k)$ for all rules F_k and the values of T are the corresponding $\mathbf{M}$ matrix. We initialize T by including adjacent matrix corresponding to all relations in the KG (e.g., $T = \{k : \mathbf{M}^{(k)}\}_{k=1}^{|R|}$).

Algorithm 2: Dynamic Programming to Calculate $\mathbf{M}$

Input: A set of chain-like definite Horn rules $\mathcal{F}$; A table $T = \{k : \mathbf{M}^{(k)}\}_{k=1}^{|R|}$

```
for F_k ∈ F do
    l ← len(B(F_k))
    for i ∈ range(l) do
        for j ∈ range(i + 1, l) do
            if B(F_k)[i : j] ∉ T then
                T[B(F_k)[i : j]] = T[B(F_k)[i : j − 1]] * T[B(F_k)[j]]
            end
        end
    end
end
```

C Analogous MAX-SAT Problem Defined Over Lukasiewicz Logic

Consider a set of logical clauses $\mathbf{C} = \{C_1, \ldots, C_m\}$, where $C_j \in C$ is a disjunction of variable $r_k(e_i, e_j)$ or its negation $\neg r_k(e_i, e_j)$, which can be written as:

$$\begin{aligned}&(\vee_{r_k(e_i,e_j)\in I_j^+} r_k(e_i, e_j))\\ \vee&(\vee_{r_k(e_i,e_j)\in I_j^-} \neg r_k(e_i, e_j))\end{aligned} \tag{7}$$

where I_j^+ (resp. I_j^-) is the set of variables that are not negated (resp. negated). Instead of interpreting the clauses $\mathbf{C}$ using Boolean logic, Lukasiewicz logic allow variables $r_k(e_i, e_j)$ to take **soft truth values** $I(r_k(e_i, e_j))$ in an interval between $[0, 1]$. Given two variables x_i and x_j, the formulas for the

relaxation of the logical conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$) are as follows:

$$I(x_i \wedge x_j) = \max\{0, I(x_i) + I(x_j) - 1\}$$
$$I(x_i \vee x_j) = \min\{1, I(x_i) + I(x_j)\}$$
$$I(\neg x_i) = 1 - I(x_i).$$

Therefore, by associating each $C_j \in C$ with weight w_j, the analogous MAX-SAT problem defined over $\mathbf{C}$ in Lukasiewicz logic can be written as:

$$\max_{\{r_k(e_i,e_j)\}\in[0,1]^n} \sum_{C_j\in\mathbf{C}} w_j \min\{ \sum_{r_k(e_i,e_j)\in I_j^+} r_k(e_i,e_j) + \sum_{r_k(e_i,e_j)\in I_j^-} (1 - r_k(e_i,e_j)), 1\} \tag{8}$$

where w_j is the weight of C_j. It is equivalent to the relaxation of MAX-SAT problem. The proof is as follows.

The MAX-SAT problem defined over weighted $\mathbf{C}$ can be formulated as the integer linear program as follows:

$$\max_{\{r_k(e_i,e_j)\}\in\{0,1\}^n} \sum_{C_j\in\mathbf{C}} w_j \min\{ \sum_{r_k(e_i,e_j)\in I_j^+} r_k(e_i,e_j) + \sum_{r_k(e_i,e_j)\in I_j^-} (1 - r_k(e_i,e_j)), 1\} \tag{9}$$

where w_j is the weight of C_j. Finding a most probable assignment to the variables $r_k(e_i, e_j)$ is NP-hard (Shimony, 1994). Using relaxation techniques developed in the randomized algorithms community, we can independently round each Boolean variable $r_k(e_i, e_j)$ to true with probability p_{ijk}. Then, the expected satisfaction score $\hat{W}$ of clauses $\mathbf{C}$ is:

$$\hat{W} = \sum_{C_j\in\mathbf{C}} w_j (1 - \prod_{r_k(e_i,e_j)\in I_j^+} (1 - p_{ijk}) \prod_{r_k(e_i,e_j)\in I_j^-} p_{ijk}) \tag{10}$$

The optimal $\hat{W}$ would give the exact MAX-SAT solution. According to (Bach et al., 2015), to approximately optimize $\hat{W}$, we can relax Eq.(9) as following:

$$\max_{\hat{\mathbf{y}}\in[0,1]^n} \sum_{C_j\in\mathbf{C}} w_j \min\{ \sum_{r_k(e_i,e_j)\in I_j^+} \hat{y_{ijk}} + \sum_{r_k(e_i,e_j)\in I_j^-} (1 - \hat{y_{ijk}}), 1\} \tag{11}$$

This results in the equivalence of Eq.(8) and MAX-SAT relaxation. Therefore, the optimum of Eq.(8) can only approximate the optimum of MAX-SAT problem.

D Satisfiability of KG Inference under Restriction of Definite Horn Rules

Given a set of logical rules $\mathcal{F}$ and their ground rules $\mathcal{F}_g$, if there exists at least one truth assignment that satisfies all ground rules $\mathcal{F}_g$, we call it *satisfiable*. We will show there exists a truth assignment to all hidden triples in a KG such that all ground rules are satisfied when restricting logical rules to be definite Horn rules.

Theorem 1. *Knowledge graph inference is satisfiable when restricting logical rules to be definite Horn rules.*

Proof. A set of ground rules is unsatisfiable if we can derive a pair of opposite ground predicates (i.e., $r_0(e_i, e_j)$ and $\neg r_0(e_i, e_j)$) from them. It is the case if and only if $\neg r_0(e_i, e_j)$ is defined in KG as definite Horn rules can only include one single positive head predicate which results in its incapability in deriving negative triples. However, a typical KG will not explicitly include negative triples (i.e., $\neg r_0(e_i, e_j)$). Thus we can never derive such a pair of opposite ground predicates, which confirms that KG inference is satisfiable when restricting logical rules to be definite Horn rules. □

E Summary of PSL-based Regularization Approaches.

PSL-based regularization methods treat logical rules as additional regularization, where satisfaction of rules is integrated into the original embedding loss. A typical integration is defined as follows: (1) sampling ground logical rules given the template logical rules; (2) mapping each related triple (i.e., predicate) into a confidence score (i.e., soft truth value); (3) computing the satisfaction score to each ground rule based on its predicates' scores; and (4) defining proper loss based on the satisfaction score for all the ground rules. We now use KALE (Guo et al., 2016) as an example to illustrate the procedure described above. First, a set of positive and negative ground rules (i.e., f^+ and f^-) are sampled given the template logical rules. Together with atomic formulas (i.e., positive and sampled negative triples), the whole set of formulas is denoted as $\mathcal{F}_g$. Second, each predicate

$r_k(e_i, e_j)$ is assigned with a soft truth value, which is a transformation of TransE-based scoring function: $I(r_k(e_i, e_j)) = 1 - \frac{1}{3\sqrt{d}} \|\mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j\|_1$, where $\mathbf{e}_i$, $\mathbf{r}_k$, and $\mathbf{e}_j$ are embedding vectors to the corresponding entities and relations and d is the dimensionality of the embeddings. Third, the soft truth value of a ground rule is computed according to Lukasiewicz Logic, where the basic operations are summarized in appendix C: Given the basic operations, the truth value of any ground formula can be calculated recursively. Finally, KALE defines a loss function over formulas from $\mathcal{F}_g$, which contain both triples and ground rules. Similar to margin-based ranking loss, embeddings are learned via maximizing the difference between the soft truth value of positive formulae $I(f^+)$ and its negative samplings $I(f^-)$. Note that, by removing ground rules from $\mathcal{F}_g$, the loss is degenerated to regular TransE-based embedding loss. Rocktäschel (Rocktäschel et al., 2015) devised a model similar to KALE. However, instead of learning entity embeddings for individual entity, they utilize matrix factorization to learn joint embeddings of pairs of entities $\mathbf{v}_{e_i,e_j}$ as well as embeddings of relations $\mathbf{v}_{r_k}$. Logistic loss is used to maximize the soft truth value of positive formulaes $I(f^+)$. Different from above methods, RUGE (Guo et al., 2017) defines a loss function over triples. It employs scoring function in ComplEx (Trouillon et al., 2016), $\sigma(\text{Re}(\langle \mathbf{e}_i, \mathbf{r}_k, \overline{\mathbf{e}}_j \rangle))$, to model triples. Triples are divided into two categories, including observed triples (i.e., v_O) and hidden triples (i.e., v_H). Observed triples have labels $y_{r_k(e_i,e_j)} = 1$ whereas sampled negative triples have labels $y_{r_k(e_i,e_j)} = 0$. The soft label of hidden triples $s_{r_k(e_i,e_j)} \in [0, 1]$ have to be predicted following t-norm fuzzy logic. With $y_{r_k(e_i,e_j)}$ and $s_{r_k(e_i,e_j)}$, RUGE learns embedding by enforcing triples to be consistent with their labels. The summary of all logical rule-based regularization approaches can be found in Table 7.

F Summary of Embedding-based Variational Inference for MLN.

To specify probability distributions over complex relational domains compactly, Markov Logic Network (MLN) (Richardson and Domingos, 2006) provides a probabilistic extension of FOL via probabilistic graphical models. Given a set of FOL formulas $\mathcal{F}$ and their corresponding weight vector w, it defines a Markov network with one node per ground predicate and one feature per ground rule. The weight of a feature is the weight of its original FOL rules. Under the MLN model, the joint probability of all triples is defined as:

$$p_w(\mathbf{v}_O, \mathbf{v}_H) = \frac{1}{Z(w)} \exp\Big(\sum_{i:F_i \in \mathcal{F}} w_i n_i(\mathbf{v}_O, \mathbf{v}_H) \Big) \tag{12}$$

where $n_i(\mathbf{v}_O, \mathbf{v}_H)$ is the number of true groundings of F_i based on the values of $\mathbf{v}_O$ and $\mathbf{v}_H$, and $Z(w)$ is a normalization constant for w to make the probabilities of all worlds sum up to one. Since MLN inference subsumes probabilistic inference, which is #P-complete, and logical inference, which is NP-complete even in finite domains (Richardson and Domingos, 2006), it is a very challenging problem computational wise. Several methods including pGAT (Harsha Vardhan et al., 2020), ExpressGNN (Zhang et al., 2019) and pLogicNet (Qu and Tang, 2019) propose to conduct variational inference of MLN to alleviate the time complexity. We now use pLogicNet (Qu and Tang, 2019) as an example to illustrate the procedure. pLogicNet aims to train the MLN model by optimizing the evidence lower bound (ELBO) for the likelihood function of observed triples $\mathbf{v}_O$:

$$\begin{aligned} \log p_w(\mathbf{v}_O) \geq & \mathcal{L}(q_\theta, p_w) \\ = & \mathbb{E}_{q_\theta(\mathbf{v}_H)}[\log p_w(\mathbf{v}_O, \mathbf{v}_H) \\ & - \log q_\theta(\mathbf{v}_H)] \end{aligned}$$

where the variational distribution $q_\theta(\mathbf{v}_H)$ is defined using a knowledge graph embedding model, by assuming each triple independently follows a Bernoulli distribution, with parameters specified by the embedding score function:

$$\begin{aligned} q_\theta(\mathbf{v}_H) &= \prod_{(e_i,r_k,e_j) \in H} q_\theta(r_k(e_i, e_j)) \\ &= \prod_{(e_i,r_k,e_j) \in H} \text{Ber}(r_k(e_i, e_j) | f_{r_k}(e_i, e_j)) \end{aligned}$$

where Ber represents the Bernoulli distribution and $f_{r_k}(e_i, e_j)$ is an embedding scoring function denoting the probability of triple (e_i, r_k, e_j) to be true. For example, in DistMult, $f_{r_k}(e_i, e_j)$ can be defined as $\sigma(\mathbf{e}_i^T \text{diag}(\mathbf{r}_k)\mathbf{e}_j)$. This lower bound can be effectively optimized using varitional EM algorithm (Neal and Hinton, 1998). In variational E-step, p_w is fixed and q_θ is updated to minimize the KL divergence between $q_\theta(\mathbf{v}_H)$ and $p_w(\mathbf{v}_H|\mathbf{v}_O)$. In M-step, q_θ is fixed and the weights of the rules w is updated to maximize the joint probability of both observed and hidden triples (i.e.,

Method	Scoring Function	Loss	Constraints
KALE (Guo et al., 2016)	$1 - \frac{1}{3\sqrt{d}} \|\mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j\|_1$	$\sum_{f^+ \in \mathcal{F}_g} \sum_{f^- \in \mathcal{N}_{f^+}} [\gamma - I(f^+) + I(f^-)]_+$	$\|\mathbf{e}\|_2 \leq 1; \|\mathbf{r}\|_2 \leq 1$
Rocktäschel et al. (Rocktäschel et al., 2015)	$\sigma(\mathbf{v}_{r_k}.\mathbf{v}_{e_i,e_j})$	$\sum_{f^+ \in \mathcal{F}_g} \mathcal{L}(I(f^+))$	ℓ_2-regularization
RUGE (Guo et al., 2017)	$\sigma(\text{Re}(\langle \mathbf{e}_i, \mathbf{r}_k, \overline{\mathbf{e}}_j \rangle))$	$\frac{1}{\|O\|} \sum_{r_k(e_i,e_j) \in \mathbf{v}_O} \mathcal{L}(I(r_k(e_i,e_j)), y_{r_k(e_i,e_j)})$ $+ \frac{1}{\|H\|} \sum_{r_k(e_i,e_j) \in \mathbf{v}_H} \mathcal{L}(I(r_k(e_i,e_j)), s_{r_k(e_i,e_j)})$	$\mathbf{e}_i, \mathbf{e}_j, \mathbf{r}_k \in \mathbb{C}^d$

Table 7: Summary of PSL-based Regularization Approaches.

$\mathbb{E}_{q_\theta(\mathbf{v}_H)}[\log p_w(\mathbf{v}_O, \mathbf{v}_H)]$). However, due to the expensive computational cost of MLN inference, even for variational inference algorithms that are developed to alleviate the time complexity, the efficiency issue remains a big problem.

G Implementation of Forward Chaining.

We have discussed the implementation of forward chaining algorithm for chain-like definite Horn rules in Section 4.1. Next, we discuss more general implementation of forward chaining algorithm to handle definite Horn rules in any form. As shown in Fig. 5, unlike traditional logical inference methods, instead of instantiating the rules with all potential triples in KG (including both observed and unobserved triples), only observed triples are considered when apply forward chaining. Let us take the definite Horn rule in Fig. 5 as an example. To apply forward chaining to infer new facts, we first focus on all observed triples related to first predicate in the body, *liveIn(person, country)*. Consequently, "country" is limited to a small set of concrete entities "USA","Denmark". By restricting "country" within the set "USA","Denmark", we again ground *officialLanguage (country, language)* with observed triples. The candidate entities for "country" will be further limited to "USA". Finally, only triples *speakLanguage (Mina Miller, English)* can be inferred as the new facts and add to the KG.

H Theoretical Computational Complexity Analysis of UniKER.

To theoretically demonstrate the superiority of our proposed UniKER in terms of efficiency, we compare the space and time complexity of UniKER and other methods that combine KG embedding and logical rules. More precisely, we only include logical rule-based regularization approaches because embedding-based variational inference for MLN is essentially a #P problem. Obviously, they have a much higher computational cost than we do. As both logical rule-based regularization approaches and our UniKER consists of two parts, materialization (i.e., sampling ground logical rules and inference U using forward chaining) and KG embedding learning, we include the complexity of both parts in Table 11. Note that materialization only contributes to the time complexity without affecting space complexity. We denote $n_e/n_r/n_t/l/n_l/\theta/a/d$ as the number of entities/relations/observed triples/length of rule body/number of rules/sampling ratio/average degree of entities/dimension of the embedding space. We can observe that: (1) For space complexity, our proposed UniKER is the same as other logical rule-based regularization approaches; (2) For time complexity, considering $a \ll n_e$, if the sampling ratio is not small enough, our proposed UniKER is much smaller than other logical rule-based regularization approaches.

I Data Statistics

The detailed statistics of three large scale real-world KGs (e.g., Family, FB15k-237 and WN18RR) are provided in Table 12. FB15K237 and WN18RR are the most widely used benchmark datasets for KGE models, which don't suffer from test triple leakage in the training set. The Family dataset is selected due to better interpretability and high intuitiveness. In addition, three small scale datasets (e.g., RC1000, sub-YAGO3-10 and sub-Family) are also included in our experiments to evaluate the scalability of forward chaining against a number of SOTA inference algorithms for MLN as shown in J.3 due to the poor scalability of MLN .

- RC1000 is a typical benchmark dataset for inference in MLN. It involves the task of relational classification with hand-code rules given.
- sub-YAGO3-10 is a subset of a well known

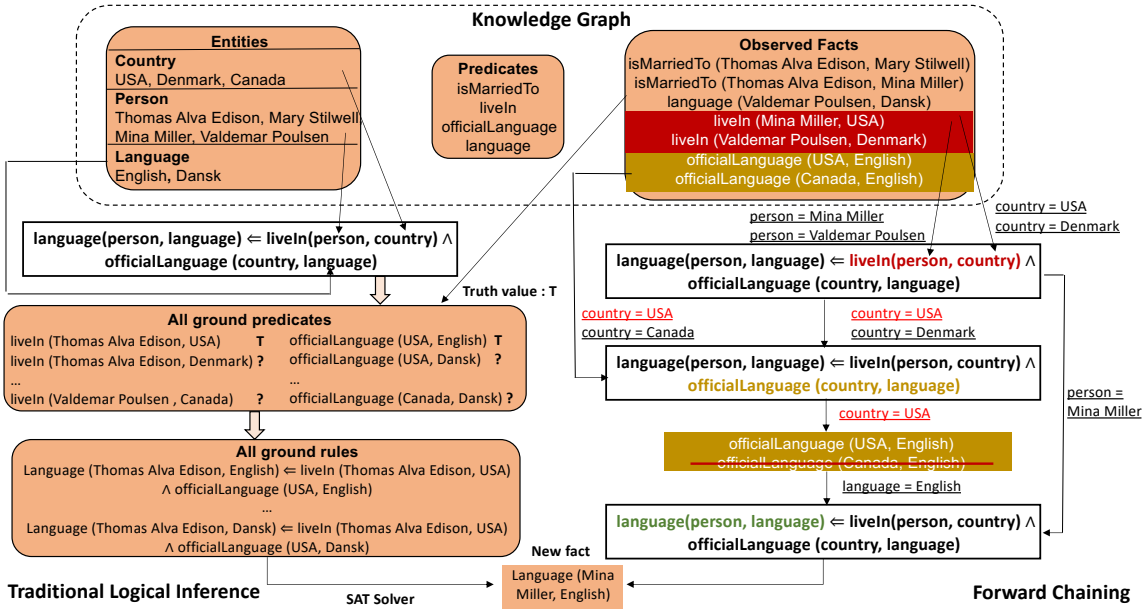

Figure 5: Traditional logical inference v.s. Forward chaining.

benchmark dataset of knowledge graph, YAGO3-10.

For the large scale knowledge graph, we adopt three commonly used benchmark datasets, including Family, FB15k-237 and WN18RR.

- Family contains family relationships among members of a family (Denham, 1973). We substract a subset from Family dataset and call it sub-Family.
- FB15k-237 is the most commonly used benchmark knowledge graph datasets introduced in (Bordes et al., 2013). It is an online collection of structured data harvested from many sources, including individual, user-submitted wiki contributions.
- WN18RR is another widely used benchmark knowledge graph datasets introduced in (Bordes et al., 2013). It is designed to produce an intuitively usable dictionary and thesaurus, and support automatic text analysis. Its entities correspond to word senses, and relationships define lexical relations between them.

J Experimental Details.

J.1 Setting for Knowledge Graph Completion

To compare among the reasoning ability of UniKER and aforementioned baseline algorithms, we mask the head or tail entity of each test triple, and require each method to predict the masked entity. We use three large-scale datasets including Family, FB15K-237 and WN18RR. During evaluation, we use the filtered setting (Bordes et al., 2013) and three evaluation metrics, i.e., Hit@1, Hit@10 and MRR. We randomly split the data into training set and test set with the ratio of 8:2 and do not exclude the triples derived from rules from test data. To fairly compare among all baseline methods, we consistently apply this same setting to all of them. Due to the unavailable codes, we take the results of BLP from the corresponding paper (Qu and Tang, 2019) and the results of pGAT from the corresponding paper (Harsha Vardhan et al., 2020). As only the results on the FB15k-237 and WN18RR datasets are reported, we only compare with them on these two datasets.

Hyperparameter Settings Adam (Kingma and Ba, 2014) is adopted as the optimizer. We set the parameters for all methods by a grid search strategy. The range of different parameters is set as follows: embedding dimension $k \in \{250, 500, 1000\}$, batch size $b \in \{256, 512, 1024\}$, and fixed margin $\gamma \in \{6, 9, 12, 24\}$. Afterwards, we compare the best results of different methods. Both the entity embeddings and the relation embeddings are uniformly initialized and no regularization is imposed on them. The detailed hyperparameter settings can be found in Table 8.

Model	Dataset	Batch Size	# Negative Samples	Embedding Dim	γ	α	Learning Rate
UniKER-TransE	FB15K-237	1024	256	1000	24	1	0.0001
	WN18RR	512	1024	500	6	0.5	0.00005
	Family	512	1024	250	12	0.5	0.001
UniKER-RotatE	FB15K-237	1024	256	250	9	1	0.00005
	WN18RR	512	1024	250	6	0.5	0.00005
	Family	1024	256	250	24	1	0.0002
UniKER-DistMult	FB15K-237	1024	256	1000	12	1	0.00005
	WN18RR	1024	1024	500	12	1	0.00005
	Family	1024	256	250	24	0.5	0.0001

Table 8: The best hyperparameter setting of UniKER on several benchmarks.

Model	ψ	Hit@1	Hit@10	MRR
UniKER-TransE	10	0.873	0.971	0.916
UniKER-TransE	20	**0.878**	0.972	**0.919**
UniKER-TransE	30	0.873	0.972	0.916
UniKER-TransE	40	0.874	**0.973**	0.917
UniKER-TransE	50	0.871	0.970	0.915

Table 9: Results of Reasoning on Family Dataset with Different Thresholds ($\psi\%$) to Include Useful Hidden Triples.

Robustness Analysis. As all kinds of noise might be contained in the process of constructing KGs, we introduce noise by substituting the true head entity or tail entity with randomly selected entity. Following this approach, we construct a noisy Family dataset with noisy triples to be 40% of original data. All generated noisy triples only fused into the original training set while validation and test sets remain the same.

J.2 Effect of Threshold Ψ Used to Include Potential Useful Hidden Triples

To investigate effect of threshold used to include useful hidden triples, we also compare the reasoning ability of UniKER with TransE on Family dataset with different thresholds Ψ. As threshold can vary a lot for different data sets, to propose a unified way to determine proper threshold Ψ, we take score $f_{r_k}(e_i, e_j)$ corresponding to the triple which ranks as top $\psi\%$ in test dataset as threshold Ψ. We vary ψ among $\{10, 20, 30, 40, 50\}$. The comparison results are presented in Table 9. We can observe that reasoning ability of UniKER does not vary a lot with different thresholds. In other words, the performance is less sensitive to the parameter ψ, which is appealing in practice.

J.3 Efficiency Analysis

We evaluate the scalability of forward chaining against a number of state-of-the-art inference algorithms for MLN (e.g., MCMC (Carlo, 2004), MC-SAT (Poon and Domingos, 2006), BP (Yedidia et al., 2001), liftedBP (Singla and Domingos, 2008) and Tuffy (Niu et al., 2011)) over all the six datasets given in Table 12. Details of each baseline are listed below:

- Markov Chain Monte Carlo (MCMC) (Carlo, 2004) is the most widely used method for approximate inference in MLN. The basic idea is to utilize Markov blanket to calculate marginal probabilities of ground predicates.
- MC-SAT (Poon and Domingos, 2006) is an efficient MCMC algorithm that combines slice sampling with satisfiability testing.
- Belief Propagation (BP) (Yedidia et al., 2001) is another widely used method for approximate inference in MLN. It is a message-passing algorithm for performing inference on graphical models.
- lifted Belief Propagation (liftedBP) (Singla and Domingos, 2008) is a lifted version of belief propagation algorithm.
- Tuffy (Niu et al., 2011) is an open-source MLN inference engine that achieves scalability compared to prior art implementations.

The results of their inference time are given in Table 13. Additionally, we compared the overall efficiency of our proposed UniKER with other methods. As shown in Table 10, UniKER showed its efficiency in the time cost per epoch. Even with the inference period, UniKER is faster than other methods combining embedding with logical rules experimentally.

Model	Time per Epoch	#Epochs for Convergence
KALE	>1000 s	500
RUGE	>1000 s	800
ExpressGNN	168 s	200
pLogicNet	7.2 s	600
UniKER-TransE	6.5 s	400

Table 10: Efficiency Analysis on Family Dataset.

J.4 Impact of Coverage of Logical Rules on Family Dataset

To further analyze the impact of coverage of logical rules on KG inference, we measure the coverage of logical rules using the total number of triples, which can be inferred from the given set of definite Horn rules. Due to space limitations, we only show the results on the Family dataset as we have similar observations on the remaining datasets. To ensure enough coverage of logical rules, we take the whole Family dataset as training data while the triples, which can be inferred from the training data using all 41 logical rules, are regarded as test data. To investigate the effects of coverage of logical rules, we vary the number of definite Horn rules among $\{10, 20, 30, 35, 36, 38, 41\}$. We provide the number of triples that can be inferred from these sets of rules in Table 14. As shown in Figure 6, we compare UniKER-DistMult against its default model DistMult as well as forward chaining algorithm. In particular, we regard link prediction in KG as binary classification and evaluate all methods in terms of triple True/False classification accuracy. We make the observations that: (1) When the coverage of logical rule is not enough, traditional rule-based methods have shown poor performance; (2) Without the incorporation of logical rules, DistMult has already shown pretty good reasoning ability; (3) The performance of UniKER sustainedly and steadily increases with the increase of coverage of logical rules; (4) When we include only 30 rules, UniKER has already achieved accuracy close to 1, which is much higher than forward chaining. A small number of logical rules is very appealing in practice as it is costly and labor-intensive to obtain high-quality logical rules.

J.5 Example of Logical Rules Used in Experiments

To generate candidate rules, we automatically mine rules using AMIE+ (Galárraga et al., 2015). Considering that longer generated rules usually suffer from lower quality and require much more computational resource, the candidate rules we used in experiments are of length at most three. Several logical rule examples for each dataset are presented in Table 15.

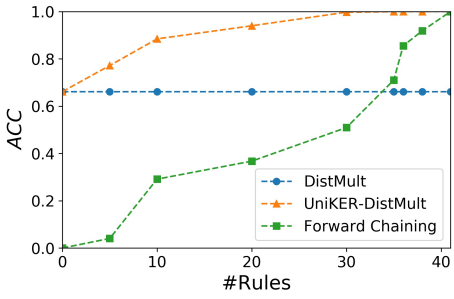

Figure 6: Impact of Coverage of Logical Rules on Family Dataset on Triple True/False Classification Task.

Method		Space Complexity	Time Complexity	
			Materialisation	Embedding
KGE	TransE (Bordes et al., 2013)	$\mathcal{O}(n_e d + n_r d)$	-	$\mathcal{O}(n_t d)$
	Dismult (Yang et al., 2014)	$\mathcal{O}(n_e d + n_r d)$	-	$\mathcal{O}(n_t d)$
Logical Rule-based Regularization	KALE (Guo et al., 2016)	$\mathcal{O}(n_e d + n_r d)$	$\mathcal{O}(\theta n_l {n_e}^{l+1})$	$\mathcal{O}(n_t d + \theta n_l {n_e}^{l+1} d)$
	Rocktäschel et al. (Rocktäschel et al., 2015)	$\mathcal{O}(n_e d + n_r d + n_e n_r)$	$\mathcal{O}(n_l n_e a^l)$	$\mathcal{O}(n_e n_r d + n_e d^2 + n_r d^2 + n_l n_e a^l d)$
	RUGE (Guo et al., 2017)	$\mathcal{O}(n_e d + n_r d)$	$\mathcal{O}(\theta n_l {n_e}^{l+1})$	$\mathcal{O}(n_t d + \theta n_l {n_e}^{l+1} d)$
Our Model	UniKER	$\mathcal{O}(n_e d + n_r d)$	$\mathcal{O}(n_l n_e a^l)$	$\mathcal{O}(n_t d + n_l n_e a^l d)$

Table 11: Comparison of Space and Time Complexity for Model Training.

Dataset	Type	#Entity	#Relation	#Triple	#Rule
RC1000	Citation network	656	4	1006	3
sub-Family	Family network	68	12	412	41
sub-YAGO3-10	YAGO knowledge	55	8	61	5
Family	Family network	3007	12	28356	41
FB15k-237	Freebase knowledge	14541	237	310116	300
WN18RR	Lexical network	40943	11	93003	11

Table 12: Data Statistics.

Model	sub-YAGO3-10	sub-Family	RC1000	Family	FB15k-237	WN18RR
MCMC	76433s	-	-	-	-	-
MCSAT	1292s	25912s	-	-	-	-
BP	10s	16343s	-	-	-	-
liftedBP	15s	16075s	-	-	-	-
Tuffy	0.849s	1.398s	4.899s	-	-	-
Forward Chaining	0.003s	0.034s	0.007s	0.593s	186s	30s

Table 13: Comparison of Inference Time for Forward Chaining vs. MLN.

#Rule	5	10	20	30	35	36	38	41
#Inferred Triple	870	6276	7915	10973	15302	18429	19780	21549

Table 14: Coverage of Different Number of Rules on Family Dataset.

Dateset	Logical Rule Examples
Family	$father(x,y) \leftarrow husband(x,z) \wedge mother(z,y)$ $nephew(x,y) \leftarrow son(x,z) \wedge sister(z,y)$ $uncle(x,y) \leftarrow brother(x,z) \wedge father(z,y)$
FB15k-237	$award\_nominations.award\_nominee(x,y)$ $\leftarrow awards\_won.award\_winner(x,z) \wedge award\_nominations.award\_nominee(y,z)$ $release\_date.film\_release\_region(x,y)$ $\leftarrow release\_date.film\_release\_region(x,z) \wedge military\_conflicts.combatants(z,y)$ $release\_date.film\_release\_region(x,y)$ $\leftarrow actor/film.performance/film(z,x) \wedge nationality(z,y)$
WN18RR	$\_also\_see(x,y) \leftarrow \_also\_see(y,x)$ $\_hypernym(x,y) \leftarrow \_verb\_group(x,z) \wedge \_hypernym(z,y)$ $\_synset\_domain\_topic\_of(x,y)$ $\leftarrow \_derivationally\_related\_form(x,z) \wedge \_synset\_domain\_topic\_of(z,y)$

Table 15: Example of Logical Rules on Family, FB15k-237 and WN18RR Datasets.