

Logical Rule-Based Knowledge Graph Reasoning: A Comprehensive Survey

Zefan Zeng , Qing Cheng * and Yuehang Si

Laboratory for Big Data and Decision, National University of Defense Technology, Changsha 410073, China; zengzefan@nudt.edu.cn (Z.Z.); siyuehang@nudt.edu.cn (Y.S.)

* Correspondence: chengqing@nudt.edu.cn

Abstract: With its powerful expressive capability and intuitive presentation, the knowledge graph has emerged as one of the primary forms of knowledge representation and management. However, the presence of biases in our cognitive and construction processes often leads to varying degrees of incompleteness and errors within knowledge graphs. To address this, reasoning becomes essential for supplementing and rectifying these shortcomings. Logical rule-based knowledge graph reasoning methods excel at performing inference by uncovering underlying logical rules, showcasing remarkable generalization ability and interpretability. Moreover, the flexibility of logical rules allows for seamless integration with diverse neural network models, thereby offering promising prospects for research and application. Despite the growing number of logical rule-based knowledge graph reasoning methods, a systematic classification and analysis of these approaches is lacking. In this review, we delve into the relevant research on logical rule-based knowledge graph reasoning, classifying them into four categories: methods based on *inductive logic programming (ILP)*, methods that unify *probabilistic graphical models* and logical rules, methods that unify *embedding techniques* and logical rules, and methods that jointly use *neural networks (NNs)* and logical rules. We introduce and analyze the core concepts and key techniques, as well as the advantages and disadvantages associated with these methods, while also providing a comparative evaluation of their performance. Furthermore, we summarize the main problems and challenges, and offer insights into potential directions for future research.



Citation: Zeng, F.; Cheng, Q.; Si, Y. Logical Rule-Based Knowledge Graph Reasoning: A Comprehensive Survey. *Mathematics* **2023**, *11*, 4486. <https://doi.org/10.3390/math11214486>

Academic Editors: Jie Wen, Yongbing Zhang and Lunke Fei

Received: 16 October 2023

Revised: 25 October 2023

Accepted: 26 October 2023

Published: 30 October 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: knowledge graph reasoning; logical rules; inductive logic programming; probabilistic graph; embedding; neural network

MSC: 68T30

1. Introduction

The big data and artificial intelligence era has brought about an abundance of knowledge-rich data. Effectively utilizing this knowledge to assist humans in problem-solving presents a significant challenge in the realm of big data. In 2012, Google introduced the knowledge graph (KG) as a straightforward yet powerful method of representing knowledge, incorporating a vast array of human knowledge resources [1]. Notably, there has been remarkable growth in open knowledge bases such as WordNet [2], DBpedia [3], YAGO [4], NELL [5], Probbase [6], CN-DBpedia [7], and zhishi.me [8], both nationally and internationally. Consequently, KGs have gained widespread adoption in facilitating practical applications of semantic networks, including recommendation systems [9], question-answering systems [10], intelligent dialogue [11], and numerous others.

There are three main categories of KGs: static knowledge graphs (SKGs), temporal knowledge graphs (TKGs), and multi-modal knowledge graphs (MMKGs), which are differentiated based on entity types and elements [12]. Among these, SKGs are the most common and extensively studied. When researchers discuss KGs without explicitly mentioning

the other two types, it is generally assumed that they are referring to SKGs. This study specifically concentrates on SKGs.

Most existing KGs suffer from limitations in knowledge sources and biases during the knowledge extraction process, resulting in incomplete or erroneous information. This necessitates the development of knowledge graph reasoning (KGR) techniques [13]. KGR is a methodology that enables the generation of new facts from existing ones within a KG. KGR has found applications in various artificial intelligence tasks, including intelligent question-answering, recommendation systems, dialogue generation, information extraction, and image classification [14–18]. Additionally, KGR plays a crucial role in domains, such as military, finance, transportation, and network security [19–22].

Despite the existence of numerous efficient KGR methods based on embedding [23,24], their utilization in latent space and the inherent black-box nature of the models present significant challenges in understanding the reasoning evidence and decision-making process in an intuitive manner. Consequently, these methods face difficulties when applied in domains that require high levels of reliability, such as military, healthcare, and information security.

Logical rule-based KGR methods directly derive new facts by extracting underlying logical rules. In addition to ensuring accurate reasoning, logical rule-based methods also exhibit strong interpretability, facilitating intuitive comprehension of the reasoning process and providing reliable guidance for decision-making. However, early logical rule-based KGR heavily relied on prior knowledge, rigid matching, and exhaustive search, resulting in limited rule expressiveness, as well as compromised scalability, robustness, and efficiency. With advancements in techniques, such as deep learning [25,26], an increasing number of studies have integrated these approaches with rule learning or utilized rules to augment relevant reasoning methods.

Large language models (LLMs) have emerged as valuable tools for knowledge reasoning [27–29]. They possess the capability to acquire extensive knowledge and utilize the abundant factual information available in pre-training datasets or external data sources for KGR tasks. However, LLMs also face potential risks associated with “hallucinations” [30], which refer to the generation of erroneous or misleading information. To address this, there has been growing interest in exploring reliable rules learning and integrating them into KGs, with the goal of enhancing LLMs through effective knowledge retrieval and augmentation. Additionally, leveraging logical rules derived from KGs to guide LLMs and developing neural-symbolic hybrid reasoning models based on LLMs can further enhance the reliability, interpretability, and domain-specific applicability of these models.

To date, several surveys on KGR have been published. Ref. [12] presents a comprehensive and detailed analysis of research on static, temporal, and multimodal KGR. Ref. [31] provides a comprehensive review of SKG reasoning, distinguishing between scenarios involving multiple samples, few-shot, and zero-shot. Ref. [32] offers a systematic introduction to KGR methods based on neural networks (NNs). Ref. [33] compiles and summarizes the key methods and technologies of graph neural networks (GNNs) applied to KGR. Ref. [1] specifically examines representation learning-based KGR. Refs. [25,26], from an interpretability standpoint, organize and classify different KGR techniques along with their respective explanation methods. Ref. [34] provides a comprehensive review of the research progress in KGR based on deep reinforcement learning (DRL). Ref. [35] classifies and introduces hybrid KGR methods, combining both symbolic and statistical approaches. Ref. [36] reviews the integration of logical rules and embedding techniques for KGR. Lastly, Ref. [37] offers a comprehensive review of symbolic, neural, and neural-symbolic hybrid reasoning methods. Compared to past works, we focus more on investigating logical rule-based KGR methods and provide a comprehensive elucidation of diverse methodologies.

The rest of this paper is organized as follows. We first provide the relevant background knowledge (Section 2). Subsequently, we comprehensively introduce logical rule-based KGR methodologies that include ILP (Section 3), methods that unify probabilistic graphical models and logical rules (Section 4), methods that unify embedding techniques and logical

rules (Section 5), as well as methods that jointly use NNs and logical rules (Section 6). We present a classification framework for these methods, visually represented in Figure 1. Our focus lies on highlighting the fundamental concepts, key techniques, and the reasoning effectiveness demonstrated in notable works within each category. Additionally, we conduct a comparative analysis to discern the distinctions among different method types (Section 7). Finally, we conclude our study by summarizing the major challenges and future prospects (Section 8). Section 9 summarizes the paper.

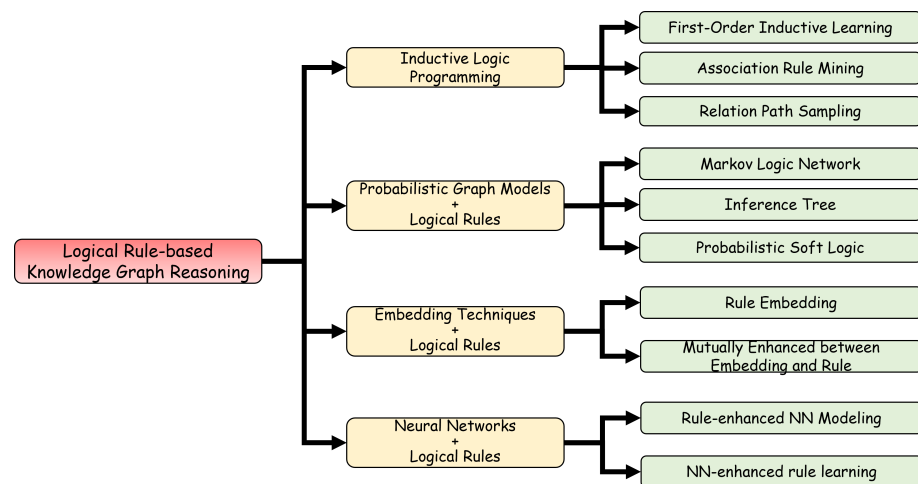


Figure 1. Taxonomy of logical rule-based KGR.

2. Background Knowledge

This section provides a comprehensive overview of the background knowledge and fundamental concepts pertaining to KG, KGR, and logical rules in KGs.

2.1. Knowledge Graph (KG)

A KG is a visual depiction of knowledge, constructed through the arrangement of nodes and edges, thereby establishing a comprehensive semantic network. Nodes symbolize concepts or entities present in the tangible realm, whereas edges connote the topological connections and semantic associations among these nodes. In [12], KGs are categorized into SKGs, TKGs, and MMKGs. TKGs include timestamp information within their structure, while MMKGs integrate diverse forms of data, such as images and text, into the various entity types they represent.

In the realm of KGs, facts are commonly expressed using triples, namely (head entity, relation, tail entity), with the vocabulary being defined within a schema (or an ontology). This schema elucidates how two entities are linked through a specific relationship. For instance, the triple (Nunez, PlayforTeam, Liverpool) denotes the fact that the player Nunez is associated with the football team Liverpool.

In the form of symbolic representation, a KG is defined as $\mathcal{KG} = (\mathcal{E}, \mathcal{R}, \mathcal{F})$, where $\mathcal{E}$, $\mathcal{R}$, and $\mathcal{F}$ denote sets of entities, relations, and facts, respectively. A triple $\langle e_h, r, e_t \rangle$ denotes a fact $\mathcal{F}$, where $e_h, e_t \in \mathcal{E}$, $r \in \mathcal{R}$, and $\mathcal{F} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. In the aforementioned triple, $e_h = \text{Nunez}$, $r = \text{PlayforTeam}$, and $e_t = \text{Liverpool}$. In the form of a graphical presentation, a KG can be regarded as a directed labeled multi-graph denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the sets of vertices and edges, respectively, with each vertex v belonging to $\mathcal{V}$ and each edge e belonging to $\mathcal{E}$. Figure 2 shows an example KG, presenting a concise depiction of its structure.

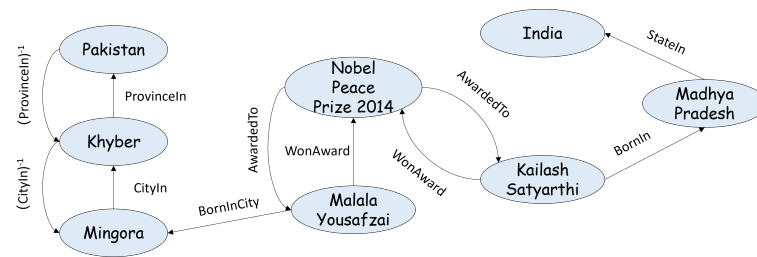


Figure 2. An example of KG.

2.2. Knowledge Graph Reasoning (KGR)

KGR refers to the utilization of specific reasoning methods to extract latent or novel facts that are implied by existing knowledge. It involves the deduction or induction of general knowledge from individual instances, aiming to enhance the comprehensiveness and accuracy of the KGs [38]. Depending on the completeness of object triples, KGR tasks can be classified into two categories: KG denoising and KG completion. The former entails assessing the accuracy of existing complete triples, while the latter focuses on predicting missing entities or relations within the KG.

In the form of symbol representation, KGR is a critical process that utilizes existing facts within $\mathcal{KG} = (\mathcal{E}, \mathcal{R}, \mathcal{F})$ to infer the queried fact (e_h^q, r^q, e_t^q) . In this context, KG denoising focuses on assessing the correctness of (e_h^q, r^q, e_t^q) , while KG completion includes three distinct subtasks: head entity completion $(?, r^q, e_t^q)$, tail entity completion $(e_h^q, r^q, ?)$, and relation prediction $(e_h^q, ?, e_t^q)$. To provide a visual representation, Figure 3 serves as an illustrative example of KGR.

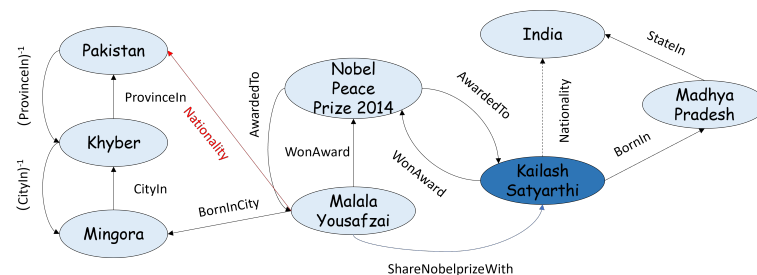


Figure 3. An example of KGR. The triples associated with the relations indicated by the red arrows pertain to the denoising task, whereas the triples indicated by the black dashed arrows are relevant to the relation prediction task. Additionally, the blue arrows and the entities they point to signify the tail entities that necessitate completion.

2.3. Logical Rule in KGs

“Logical rule” is commonly referred to as “rule”. The fundamental unit of a logical rule in KGs is referred to as an *atom*, conventionally denoted by the symbol α . An atom can be defined as a unary or n -ary (typically binary) predicate symbol that maps a set of variables to a Boolean value. Notably, when all variables in an atom are constants, it is considered a *grounding atom*. A logical rule is composed of connecting multiple atoms using logical connectors. Typically, rules are formulated in the form of *body* $\rightarrow$ *head*, where the body consists of a conjunction of multiple atoms denoting the antecedent or conditions of the rule, and the head is a single atom containing the target predicate that signifies the conclusion derived from the rule.

First-order logic (FOL) [39] serves as a fundamental approach for knowledge representation and reasoning. It facilitates the transformation of triples within a KG into combinations of “binary predicates + constants,” exemplified by the forms (subject, predicate, object) and predicate(subject, object). For instance, in Figure 3, the rule $\text{BornInCity}(X,Y) \cap \text{CityIn}(Y,Z) \cap \text{ProvinceIn}(Z,W) \rightarrow \text{Nationality}(X,W)$ enables us to deduce that Malala Yousafzai’s nationality is Pakistani.

Horn rule logic [40], also denoted as Horn rules, is a specialized subset of FOL. It is distinguished by its simplicity and the ease with which it can be represented, rendering it a versatile method for rule representation. A Horn rule comprises a head, which is a singular atomic formula, and a body, which is a conjunction of atomic formulas. A typical expression of a Horn rule is as follows:

$$H : A(\alpha_1, \dots, \alpha_n) \rightarrow \alpha, \quad (1)$$

here α denotes the head atom, while $\alpha_1, \dots, \alpha_n$ ($n > 0$) denote the body atoms. The rule body, denoted as A , includes the body atoms that are connected using logical paradigms such as conjunction or disjunction. Referred to as the formula, the rule body dictates that if A holds true, then the head atom of the rule also holds true. *Closed rules* within Horn rule logic pertain to those where every variable appears in at least two atoms. These closed rules are widely employed in KGR.

Logical rule-based KGR represents a generalization of rule paradigms that conforms to the semantics of natural language expressions and human thought processes. This approach offers explicit and transparent reasoning foundations, thereby ensuring robustness and interpretability in reasoning. However, the diverse semantic nature of KGs often entails intricate rules in reasoning, posing challenges for traditional rule learning and KGR methods. Notably, these approaches exhibit limitations in terms of scalability, expressive power, as well as time and space complexity. To tackle these challenges, researchers have explored the integration of graph models, NNs, and other techniques with rules. Leveraging the inherent advantages of transferability, robustness, and efficiency, these hybrid reasoning approaches aim to surmount the aforementioned limitations.

3. Inductive Logic Programming-Based KGR

ILP is a rule-learning technique derived from FOL [37]. The primary objective of ILP is to uncover hidden patterns within data by identifying logical programs, rules, or formulas that are shared across the dataset. This method imposes constraints on logical rules, specifically restricting them to Horn rules. By utilizing given predicates, background atoms, positive examples, and negative examples, ILP constructs rules capable of elucidating positive examples while discarding negative ones. KGR approaches leveraging ILP can be classified into three categories: methods based on first-order inductive learning, association rule mining, and relation path sampling. Table 1 summarizes the relevant information of ILP-based KGR methods.

Table 1. Summary and comparison of ILP-based KGR methods.

Category	Method	Year	Need Negative Examples	Traversing Search	Fuzzy Reasoning
FOIL	FOIL [41]	1990	✓	✓	×
	N-FOIL [42]	2007			
	T-FOIL [42]	2007			
	KFOIL [43]	2010			
	QuickFOIL [44]	2015			
Association Rule Mining	AMIE [45]	2013	×	✓	×
	AMIE+ [46]	2015			
	RDF2Rules [47]	2015			
	AMIE3 [48]	2020			
Relation Path Sampling	RuleN [49]	2018	×	×	×
	AnyBURL [50]	2019			
	C-NN [51]	2019			
	Reinforce-AnyBURL [52]	2020			
	SAFRAN [53]	2021			

3.1. First-Order Inductive Learning-Based KGR

First-order inductive learner (FOIL) is a rule-learning technique that is agnostic to specific domains [41]. Its primary objective is to acquire inference rules from textual data. FOIL builds upon the SHERLOCK system [54] and follows a four-step process to learn rules: (1) identification of relevant classes and their corresponding instances, (2) discovery of inter-class relations, (3) acquisition of inference rules based on these relations and determination of their confidence scores, and (4) utilization of the HOLMES engine to perform logical reasoning and search for facts, with the confidence of each fact estimated using a Markov network. Please refer to Figure 4 for an illustration of this process.

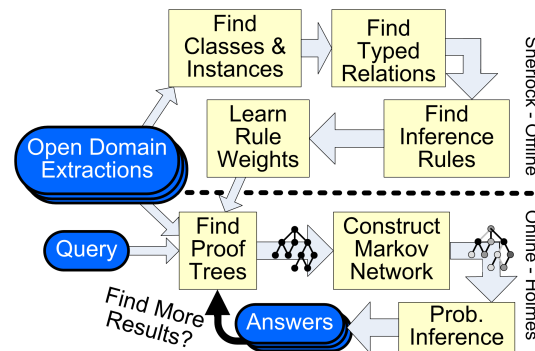


Figure 4. Architecture of FOIL. SHERLOCK learns logical rules offline and provides them to the HOLMES inference engine, which uses the rules to answer queries.

Landwehr et al. [42] proposed two dynamic propositionalization rule-learning methods, N-FOIL and T-FOIL, by integrating the naive Bayes learning method into FOIL. N-FOIL incorporates a mechanism to reduce the number of instances in each iteration. Moreover, it introduces an adaptive scoring path for extra rules to enhance the diversity of rule learning effectively. To mitigate the issue of excessive model parameters, N-FOIL assumes that the probabilities of instances satisfying different queries are mutually independent (Equation (2)), namely

$$P_{\lambda}(q_1, \dots, q_k | p) = \prod_i P_{\lambda}(q_i | p), \quad (2)$$

P_{λ} denotes the probability of a query, p denotes the class label, and q_i denotes the query. However, in practical scenarios, it is often difficult to satisfy this assumption. To overcome this challenge, T-FOIL proposes a solution by introducing an additional dependency tree, which effectively reduces the number of parameters involved. T-FOIL aims to address the issue and enhance the practical applicability of the model. Thus, Equation (2) is relaxed to the following:

$$P_{\lambda}(q_1, \dots, q_k | p) = \prod_i P_{\lambda}(q_i | p, q_{pa(i)}), \quad (3)$$

where $pa(i)$ denotes the parent nodes associated with the node q_i . By incorporating a tree-augmented strategy, T-FOIL reduces the number of model parameters from $\mathcal{O}(2^{\#nodes})$ to $\mathcal{O}(\#nodes)$, then reduces the computational complexity.

K-FOIL [43] integrates kernel learning techniques with the FOIL algorithm. In the K-FOIL approach, structure learning infers an appropriate kernel for logical objects, while parameter learning aims to reproduce function learning within the Hilbert space of the kernel. To induce the kernel function K within the logical setting, a discrete space search algorithm is employed, which enables multi-task statistical logical rule learning using K . Compared to FOIL and N-FOIL, K-FOIL demonstrates advantages in terms of both accuracy and efficiency.

QuickFOIL [44] adopts a top-down greedy ILP search and integrates a scoring function and pruning strategy to guide the heuristic search process, thereby facilitating the discovery of high-quality rules. During the construction of sub-rules, QuickFOIL initializes with an

empty clause and utilizes a greedy heuristic approach to incrementally add new clauses while selectively pruning crucial decisions in the search space. QuickFOIL effectively reduces the number of join operations as well as incorporating diverse query optimization and caching techniques to expedite query processing speed.

Limitations. FOIL-based approaches have been proven effective in performing efficient reasoning within small-scale KGs. However, when dealing with medium and large-scale KGs, the proliferation of entity and relation types introduces a substantial increase in the number of rules, resulting in exponential growth of computational complexity. Consequently, the accuracy and efficiency of these methods are significantly compromised. Furthermore, FOIL methods commonly rely on the availability of negative examples, yet the open-world assumption (OWA) prevalent in KGs renders the use of missing data as negative examples unfeasible.

3.2. Association Rule Mining-Based KGR

To overcome the limitations of the FOIL-based methods, researchers proposed association rule mining algorithms. These approaches leverage existing knowledge or employ sophisticated search algorithms to construct a repository of rules. Subsequently, they traverse the KGs, seeking triples that align with the candidate rules.

AMIE [45] is a system devised to mine association rules with incomplete evidence in large-scale knowledge bases. It constructs a rule repository by considering the relation types present in a KG, and then identifies instances that satisfy these rules. A rule is deemed reliable if its confidence score surpasses a pre-defined threshold. The rule-learning process in AMIE comprises two stages. In the first stage, rules are treated as atomic sequences, and an empty rule set is initialized. Three mining operators are employed to add rules:

- Dangling atoms $\mathcal{O}_D$ are added by introducing new variables to existing rules,
- Instantiated atoms $\mathcal{O}_I$ are added by incorporating constant atoms into existing rules,
- Closed atoms $\mathcal{O}_C$ are added by introducing a new atom that does not expand the rule further.

The second stage involves rule pruning, wherein the partial completeness assumption (PCA) is employed to compute confidence scores. If adding an atom to a rule fails to improve its confidence score, the rule is no longer expanded. Additionally, a rule that solely consists of a head relation is evaluated against a threshold for its confidence score. If the threshold is not surpassed, the rule is directly discarded.

AMIE operates without the need for external knowledge beyond the KG and does not necessitate configuration or parameter tuning. It demonstrates remarkable enhancements in terms of runtime, the number of output rules, and rule quality compared to FOIL-based methods.

AMIE employs various constraints, including rule length, type, and confidence, to reduce the search space. However, when dealing with large-scale KGs, the rule discovery process suffers from a combinatorial explosion due to the unrestricted connection between all relations. To tackle these challenges, AMIE+ [46] focuses on three key optimizations:

- During the rule discovery step, it introduces $\mathcal{O}_C$ and $\mathcal{O}_I$ in the final stage, avoiding the addition of $\mathcal{O}_D$ to prevent rule non-closure.
- The rule pruning step incorporates a confidence approximation method to reduce computational complexity.
- Rule discovery is enriched by incorporating heuristic-type checks and joint reasoning.

AMIE+ is limited to mining one rule at a time, precluding parallelization. To overcome this setback, AMIE3 [48] integrates several sophisticated pruning strategies, parallelization optimization algorithms, and a lazy computation approach for confidence estimation to expedite rule mining. AMIE3 is specifically designed for large-scale KGs and can comprehensively mine all rules satisfying the specified support and confidence thresholds, without relying on sampling or approximation techniques.

RDF2Rules [47] employs frequent predicate cycles (FPC) for association rule generation. In RDF2Rules, a predicate cycle is defined as a sequential connection of predicates that form a loop among entities, denoted as $\{x_1, p_1 d_1, x_2, \dots, p_k d_k, x_1\}$, where x_i denotes an entity, p_i denotes a predicate, and $d_i \in \{-1, 1\}$ denotes the direction of the predicate. Initially, the FPC is identified through support calculation. Subsequently, the FPC is utilized to generate and evaluate rules, leveraging entity-type information to improve the precision of the learned rules.

Limitations. Despite the advancements in efficiency compared to FOIL-based methods, association rule mining approaches often necessitate exhaustive traversal of all relation paths. When applied to large-scale KGs, these methods encounter high computational complexity.

3.3. Relation Path Sampling-Based KGR

To mitigate the computational complexity associated with rule searching, scholars introduced KGR methods based on relation path sampling. The fundamental concept behind this approach entails sampling a subset of triples from the KG that are directly linked to the target relation, and leveraging this sampled information for rule learning.

RuleN [49] is an advanced rule mining technique that leverages dual random sampling. It facilitates two distinct types of rule mining, namely P_n -type rules and C-type rules:

$$P_n : r(x_1, x_{n+1}) \leftarrow s_1(x_1, x_2) \wedge \dots \wedge s_n(x_n, x_{n+1}), \quad (4)$$

$$C : r(x, a) \leftarrow \exists y r(x, y). \quad (5)$$

For the P_n -type relation r , only K triples (a, r, b) that are associated with it are extracted. A depth-first search algorithm is employed to identify all paths of length n between a and b as candidate rule bodies. These paths are then randomly sampled to determine if there exists a relation r between the head entity and tail entity. For the C-type rule, a similar process is followed where K triples (a, r, b) are randomly selected. For each triple, rules of the form $r(x, b) \leftarrow r(x, y)$ and $r(a, y) \leftarrow r(x, y)$ are created. A randomly sampled triple that conforms to the rule is then used to calculate the approximate confidence score of the rule. By reducing the search space through sampling, this method simplifies the computational process.

C-NN [51] leverages entity clustering to generate rules without requiring extensive training or retraining, thereby mitigating the computational costs associated with runtime and memory. Notably, C-NN employs a symbolic representation of K-nearest neighbors, wherein the similarity between entities is quantified by computing distances on a graph. To identify clusters of similar entities, neighbor selection sampling is employed for each entity. From these clusters, diverse rules are derived, facilitating joint reasoning and providing comprehensive explanations for each inference.

AnyBURL [50] is a bottom-up technique that facilitates the on-demand acquisition of logical rules from extensive KGs. AnyBURL adopts a sampling-based approach, starting from $n = 2$, wherein paths of length n are sampled from a given KG, and rules of length $n - 1$ are inferred from these paths. The initial edge on each path represents the head atom. As the saturation threshold for rules of length $n - 1$ is exceeded, the algorithm incrementally increases the value of n to learn longer rules. This rule-learning process operates within a predefined time span, during which the algorithm iteratively samples random paths to maximize rule acquisition. Upon the conclusion of the time span, the rules are evaluated and assigned confidence scores. Finally, candidate rules are ranked based on the maximum confidence score derived from all candidates.

AnyBURL boasts superior computational efficiency and minimal resource requirements. Moreover, it facilitates the generation of explanations by leveraging the proposed candidate rules.

The effectiveness of the rules generated by AnyBURL heavily depends on the appropriate setting of the saturation threshold. However, determining an optimal threshold is

challenging. To tackle this problem, Reinforce-AnyBURL [52] leverages DRL by treating interpretability, confidence, and rule length as reward factors during the rule-learning process. Moreover, an object identification method is introduced to impose constraints on the rules, thereby eliminating redundant and duplicate rules.

SAFRAN [53] introduces a novel strategy for non-redundant noise aggregation, which involves detecting and clustering redundant rules prior to aggregation. Hence, the negative influence of redundant information is alleviated, leading to improved prediction performance. SAFRAN adopts a maximum aggregation method to predict confidence scores for entities, followed by additional aggregation based on noise for distinct prediction clusters. Furthermore, SAFRAN employs both grid search and random search techniques to determine the optimal clustering threshold for rule aggregation based on type combinations. In comparison to AnyBURL, SAFRAN achieves superior rule mining quality.

3.4. Open Challenges

ILP-based rule-learning and KGR methods are categorized as “hard” reasoning approaches. In these methods, the validity of the discovered rules is determined by evaluating them against a threshold through the use of confidence scores. However, it is important to note that these methods are mostly subjective in threshold settings. Moreover, their applicability is constrained to precise reasoning, lacking the ability to represent uncertainty information.

4. Unifying Probabilistic Graph Models and Logical Rules for KGR

To facilitate fuzzy reasoning, scholars integrated probabilistic graph models with rule learning to introduce a range of “soft” reasoning methodologies that are well-suited for fuzzy inference. Notably, these methodologies include diverse approaches such as methods based on a Markov logic network, inference trees, and probabilistic soft logic. Table 2 summarizes the relevant information of these methods.

Table 2. Summary and comparison of KGR methods that unify probabilistic graph models and logical rules.

Category	Method	Year	Work in KGs with Missing Triples	Work in Large-Scale KGs	Traversing Search
MLN [55]		2005	×	×	✓
Inference-Tree	ProbLog [56]	2007			
	SLP [57]	2001	✓	×	✓
	ProPPR [58]	2013			
Probabilistic Soft Logic	PSL [59]	2012			
	OAP [60]	2013	✓	✓	✓
	HL-MRF [61]	2017			

4.1. Markov Logic Network

The Markov logic network (MLN) [55] establishes a probabilistic graphical model by leveraging predefined rules and factual information extracted from the KG. Subsequently, the network undergoes a weight learning process, where weights associated with various rules are adjusted accordingly.

Given an initial set of rules $\{\gamma_i\}$, the modeling process of MLN entails the following key steps: (1) Basic atoms of each rule are mapped to individual nodes. The value of the node is assigned as 1 if the atom exists in the KG; otherwise, it is set as 0. (2) If two basic atoms have the potential to instantiate at least one rule simultaneously, an edge is established between their corresponding nodes. (3) The collection of all basic atoms forms a clique, representing a specific feature. The feature’s value is 1 when the basic rule holds true; otherwise, it is 0.

As shown in Figure 5, the dashed lines in the figure correspond to cliques associated with γ_1 , while the solid lines represent cliques associated with γ_2 . By taking into account γ_1 , γ_2 , and the observed basic atoms, as well as the unobserved basic atom $\text{Lives_in}(\text{LeBron}, \text{L.A.})$, an MLN can be constructed. Firstly, an edge is established between $\text{Lives_in}(\text{LeBron}, \text{L.A.})$ and $\text{Spouse_of}(\text{LeBron}, \text{Savannah})$ since they can be simultaneously utilized for instantiating γ_1 . Subsequently, $\text{Lives_in}(\text{LeBron}, \text{L.A.})$, $\text{Spouse_of}(\text{LeBron}, \text{Savannah})$, and $\text{Lives_in}(\text{Savannah}, \text{L.A.})$ form a clique as they constitute the basic atoms of the rule γ_1 . The same approach can be employed to construct cliques for the rule γ_2 .

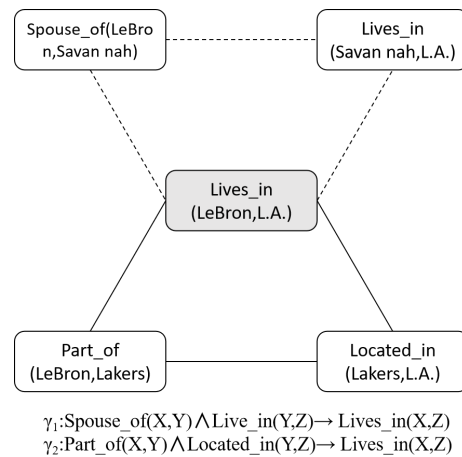


Figure 5. Two examples of rules and the corresponding ground MLN (dotted lines are clique potentials associated with rule γ_1 , and solid lines are with rule γ_2 . The gray node is the unobserved triple to be reasoned).

In MLN, the joint distribution of the values X for all nodes (atoms) is defined as:

$$P(X = x) = \frac{1}{Z} \exp \left(\sum_i w_i n_i(x) \right), \quad (6)$$

$n_i(x)$ denotes the number of true basic atoms in the rule γ_i , and w_i denotes the weight associated with the rule γ_i . Subsequently, the Markov chain Monte Carlo (MCMC) algorithm is applied within MLN for inference, and the weights w_i are effectively learned by optimizing the pseudo-likelihood measure. By utilizing the calculated values of w_i and considering the neighboring basic atoms, we can infer the confidence score of $\text{Lives_in}(\text{LeBron}, \text{L.A.})$.

Limitations. MLN is the pioneering model employed for fuzzy reasoning within KGs. Nonetheless, the inference procedure of MLN encounters difficulties and inefficiencies attributable to the intricate graph structure present among the triples. Moreover, the missing data of certain triples in KGs can further influence the inference results via rules.

4.2. Inference Tree-Based KGR

Selective linear definite (SLD) [56] emerges as a methodology to conduct KGR through the construction of inference trees. In contrast to MLN, SLD inference trees exhibit proficiency in managing intricate triple structures, thus yielding enhanced inference efficiency. SLD assembles the inference tree in a top-down fashion, as shown in Figure 6. It commences by initializing the root node using the query and subsequently generates sub-goals recursively by employing individual clauses alongside their respective instances.

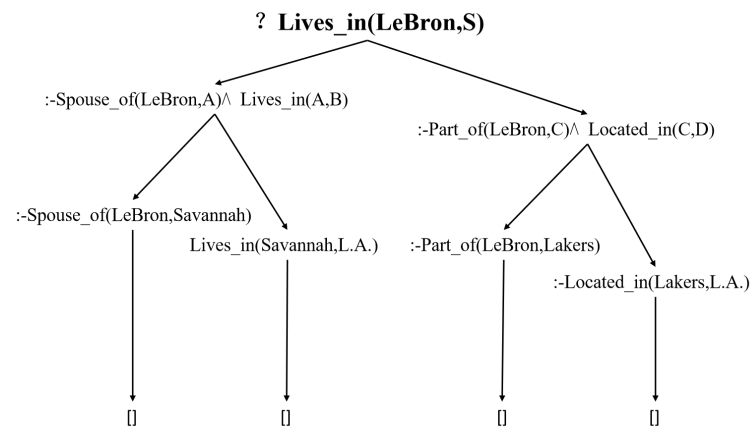


Figure 6. An example of the SLD-tree built for the query $Lives_in(LeBron, S)$.

Based on SLD inference trees, ProbLog [56] is a probabilistic extension of the Prolog logic programming approach [62]. ProbLog introduces a probability p_i for each clause c_i , where c_i denotes a rule or a basic atom used to derive examples for a given query. The final inference result is obtained through recursive reasoning based on the constructed inference tree. Notably, in ProbLog, all rules or atoms are assigned probabilities. The probability distribution of the rule set is denoted as $T = \{p_1 : c_1, \dots, p_n : c_n\}$, where $L \subseteq L_T = \{c_1, \dots, c_n\}$ denotes a subset of the rule set. Given a query q , the probability $P(q|T)$ of successfully inferring the answer from the rule set can be decomposed as follows:

$$P(q|T) = \sum_{L \subseteq L_T} P(q, L|T) = P(q|L)P(L|T) = P(q|L) \left[\prod_{c_i \in L} p_i \prod_{c_i \in L_T \setminus L} (1 - p_i) \right], \quad (7)$$

$P(q|L)$ denotes the probability of successfully inferring the query q given a subset of rules L . If there exists at least one answer that can instantiate L and satisfy the query, $P(q|L)$ is assigned a value of 1; otherwise, it is assigned a value of 0. The calculation of $P(q|L)$ involves constructing an SLD tree for the query q in ProbLog. The process begins by initializing the root node based on the query and proceeds recursively by creating sub-goals through the application of each clause and its instantiation, until a termination condition is met (such as finding an answer or reaching the maximum tree depth).

To enhance the efficiency of the inference tree, SLP [57] incorporates parameterized clauses and utilizes a logarithmic-linear distribution to invert the goal. SLP defines a stochastic process that traverses the SLD tree, where the probability distribution at each node is determined by assigning higher weights to the required answer clauses and lower weights to the other clauses. Furthermore, a fault-adjusted maximization (FAM) method is developed to further improve the speed of rule optimization. FAM provides a closed-form solution for calculating parameter updates, thereby enabling efficient rule optimization.

ProPPR [58] employs a biased sampling strategy to replace the random sampling strategy used in SLP. Unlike directly assigning probabilities to each query as in Equation (7), ProPPR computes the probability for each clause by leveraging predefined features from the PPR algorithm. Additionally, a local grounding process is devised to facilitate inference. This local grounding process has the ability to divide the rule weight learning task into independent gradient computations, enabling parallel learning and enhancing the efficiency of rule learning and reasoning.

In addition to ProbLog, SLP, and ProPPR, Datalog [63] and MV-Datalog $\pm$ View [64] also utilize SLD for knowledge reasoning.

Limitations. While inference tree-based methods have shown advancements in terms of speed compared to MLN, they still fall short of achieving satisfactory reasoning accuracy and efficiency when confronted with KGs that entail a substantial number of missing triples.

4.3. Probabilistic Soft Logic-Based KGR

To tackle the reasoning challenge posed by missing triples in KGs, Probabilistic soft logic [59] offers a framework for collective probabilistic reasoning in relational domains, leveraging Boolean logical rules. It comprises a collection of FOL rules, wherein bodies are interconnected and heads are individual literals.

In PSL, the tilde operator is utilized to express logical relaxation within the Boolean domain. For a fundamental rule denoted as $r \equiv r_{body} \rightarrow r_{head} \equiv \sim r_{body} \tilde{\vee} r_{head}$, the fulfillment of r relies on the confidence score of its constituent atoms, denoted as I (AKA true value). To evaluate the level of satisfaction with these basic rules, PSL makes use of the Łukasiewicz t-norm [65] and its corresponding co-norm (Equation (8)) $d_r = \max\{0, I(r_{body}, r_{head})\}$ to compute the “gap” between the rule and its satisfaction:

$$\ell_1 \tilde{\vee} \ell_2 = \min\{I(\ell_1) + I(\ell_2), 1\}, \quad (8)$$

$$\sim \ell_2 = 1 - I(\ell_1), \quad (9)$$

where ℓ denotes an atom. By employing Equation (8), the Boolean truth values of rule confidence are relaxed, resulting in continuous soft truth values within the interval $[0, 1]$.

PSL integrates similarity functions into the graphical model through the incorporation of relaxation and distance calculation. Additionally, PSL imposes syntax restrictions on first-order formulas, specifically limiting them to rules with connected bodies. These two characteristics facilitate the transformation of reasoning into a convex optimization problem in continuous space, thereby enabling convenient rule learning and solving even in situations where knowledge is incomplete.

To reduce the time required for recognition and reasoning in KGs, as well as to overcome reasoning noise, OAP [60] constructs an ontology KG, illustrated in Figure 7. PSL is employed here to establish a joint probability distribution over the KG. Subsequently, the minimum-cut technique is applied to partition the relations and labels present in the ontology, thereby creating distinct partitions for the extraction of relation and label-specific instances that facilitate knowledge reasoning.

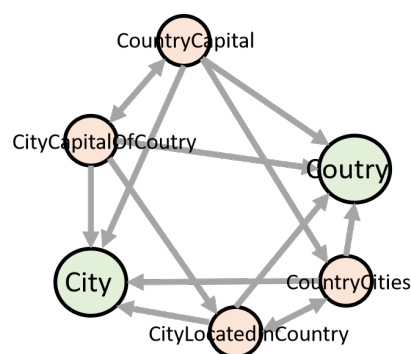


Figure 7. An ontology knowledge graph constructed by the OAP. Entities and relations serve as nodes and ontology-constrained relations act as edges

HL-MRF [61] integrates stochastic algorithms, probabilistic graph models, and fuzzy logic communities. HL-MRF leverages the connection between Hinge-Loss potentials and logical rules to define PSL, enabling the model to efficiently handle large-scale KGs. Furthermore, PSL provides a syntax that enhances the interpretability and facilitates its extensibility by users.

4.4. Open Challenges

While methods that unify probabilistic graph models and logical rules have the ability to perform fuzzy reasoning, they are fundamentally dependent on rule-based search mining, which requires complex traversal computations on the KG. As a result, their accuracy, efficiency, and robustness are not well-suited for large-scale KGs.

5. Unifying Embedding Techniques and Logical Rules for KGR

Traditional KGR methods, which rely on explicit rule learning, are known for their interpretability and explainability. However, these methods face limitations as the discrete and limited rules struggle to capture the entirety of the intrinsic relations within the data. Additionally, they lack robustness when dealing with fuzziness and noise. KG embedding (KGE) represents entities and relations using dense vectors in a low-dimensional continuous vector space. KGE allows for effective representation and measurement of semantic associations between entities and relations. As a result, by unifying embedding techniques and logical rules, computational efficiency is improved, data sparsity is alleviated, and robustness to fuzziness and noise is enhanced.

To evaluate the correctness probability (truth value) for a given triple, the embedding-based learning of a scoring function is utilized. Extensive research has been conducted to explore the integration of embedding and rule learning for reasoning. Such approaches can be categorized as rule embedding-based KGR and mutual enhancement between embeddings and rules for the KGR. Table 3 summarizes the relevant information of these approaches.

Table 3. Summary and comparison of KGR methods that unify embedding techniques and logical rules.

Category	Method	Year	Model Entities and Relations as	Need Prior Rules	Sparse KGR
Rule Embedding	KALE [66]	2016	Vector	✓	×
	ProPPR+MF [67]	2016	Matrix	✓	×
	TARE [68]	2017	Vector/Matrix	✓	×
	RUGE [69]	2018	Vector	✓	×
	Com-NN-ER [70]	2018	Vector/Matrix	×	×
	Wang et al. [71]	2019	Vector/Matrix	✓	×
	RPJE [72]	2019	Vector	✓	✓
	RuE [73]	2023	Vector	✓	×
Mutually Enhanced between Embedding and Rule	RuLES [74]	2018	/	✓	✓
	IterE [75]	2019	Vector/Matrix	×	✓
	RLvLR [76]	2019	Matrix	×	×
	UniKER [77]	2021	Vector/Matrix	✓	✓

5.1. Rule Embedding-Based KGR

These methods facilitate reasoning by utilizing either the embedding or the enhanced embedding of rules. Rules provide valuable information for KGE. Additionally, the triples derived from rule inference can be employed as supplementary training samples for embedding, resulting in enhanced efficiency and quality of the embedding. Importantly, the incorporation of rules also contributes to the provision of additional explanatory evidence for reasoning results, thereby augmenting the interpretability of the process.

Modeling rules as vectors. KALE [66] incorporates both rules and triples through joint embedding. Within this approach, triples are considered as fundamental atomic units, while logical rules are instantiated with specific entities, treating them as intricate formulas composed of basic atoms and logical connectors. To model the triples, TransE [78] is employed, where the truth value I is defined as:

$$I(e_i, r_k, e_j) = 1 - \left\| \frac{1}{3\sqrt{d}} (\mathbf{e}_i + \mathbf{r}_k - \mathbf{e}_j) \right\|, \quad (10)$$

d is the embedding dimension. In the context of rule modeling, a fuzzy logic approach is employed, which considers two types of rules. The first type is represented as $\forall x, y : (x, r_s, y) \rightarrow (x, r_t, y)$, and it is denoted as $f_1 \triangleq (e_m, r_s, e_n) \rightarrow (e_m, r_t, e_n)$. The second

type is represented as $\forall x, y, z : (x, r_{s1}, y) \wedge (y, r_{s2}, z) \rightarrow (x, r_t, z)$, and it is denoted as $f_2 \triangleq (e_l, r_{s1}, e_m) \wedge (e_m, r_{s2}, e_n) \rightarrow (e_l, r_t, e_n)$. The computation equation for evaluating the truth value I of these two rules is described as follows:

$$\begin{aligned} I(f_1) &= I(e_m, r_s, e_n) \cdot I(e_m, r_t, e_n) - I(e_m, r_s, e_n) + 1, \\ I(f_2) &= I(e_l, r_{s1}, e_m) \cdot I(e_m, r_{s2}, e_n) \cdot I(e_l, r_t, e_n) - I(e_l, r_{s1}, e_m) \cdot I(e_m, r_{s2}, e_n) + 1, \end{aligned} \quad (11)$$

the larger the value of I , the higher the likelihood of obtaining correct triples through reasoning.

Modeling rules as matrices. Different from KALE, ProPPR-MF [67] leverages matrix factorization for embedding FOL. By using transferable matrix factorization and low-rank approximation, the method acquires latent continuous representations of instances and logical rules. The detailed steps of ProPPR-MF are illustrated in Figure 8. By learning embeddings of logical rules, discrete logical facts and predicates are transformed into continuous matrix vectors. This enhancement enables its application to expansive and intricate KGs.

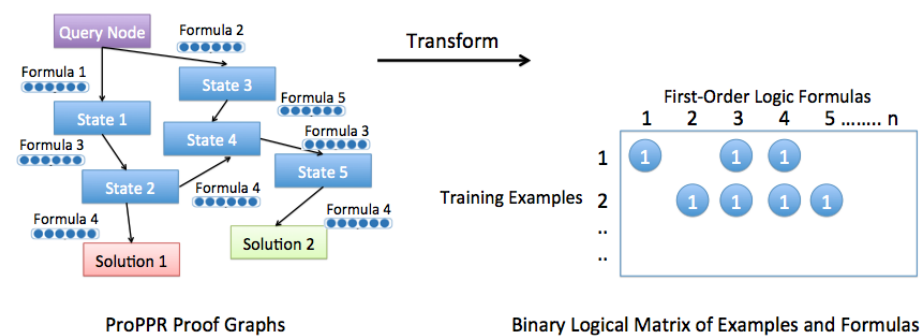


Figure 8. ProPPR-MF for learning FOL embeddings. Initially, ProPPR-MF devises a structural learning strategy based on structural gradients to generate plausible inference rules from factual information. Subsequently, it constructs comprehensible proof graphs using background facts, training examples, and inference formulas. To facilitate this process, training examples are mapped to rows of a binary matrix, while inference formulas are mapped to columns of another binary matrix.

Iterative rule injection and soft rule learning. KALE and ProPPR-MF employ a one-time injection of logical rules, disregarding the interaction between embedding learning and logical reasoning. Additionally, they primarily focus on hard rules, which often entail substantial manual effort for creation or validation.

To address the above limitations, RUGE [69] incorporates soft rule guidance and replaces the single-round rule injection with an iterative process. To extract rules, RUGE utilizes AMIE without directly considering basic rules as positive instances. Instead, it treats the triples derived from rules as unlabeled triples and employs observable triples to update embeddings. Subsequently, RUGE predicts the probability of each unlabeled triple based on the current embeddings. Finally, the embeddings are updated using both labeled and unlabeled triples. ComplEx [79] is utilized for embedding triples.

One-to-one rule mapping embedding. RUGE employs a scoring mechanism based on the subarrays derived from the decomposition of rules or formulas. However, the individual calculation of triple scores can result in inflated scores for rules or formulas, even when the triples within them are unrelated.

To address the above problem, Wang et al. [71] introduce a conversion process where a triple or basic rule is transformed into FOL. Subsequently, FOL is evaluated by performing vector or matrix operations using the embeddings of the entities and relations involved. The format of the FOL and the scoring method are presented in Table 4 and the accompanying mathematical expressions are presented in Table 5. Through this conversion, distinct triples within the same rule directly interact in the vector space, ensuring a one-to-one mapping transformation for both the rule and its encoding format.

Table 4. The corresponding format for triples and their ground rules.

Triples and Ground Rules	Format
(h, r, t)	$r(h) \Rightarrow t$
$(h, r_1, t) \Rightarrow (h, r_2, t)$	$[(h \in C) \wedge [r_1(h) \Rightarrow t]] \Rightarrow [r_2(h) \Rightarrow t]$
$(e_1, r_1, e_2) + (e_2, r_2, e_3) \Rightarrow (e_1, r_3, e_3)$	$[[r_1(e_1) \Rightarrow e_2] \wedge [r_2(e_2) \Rightarrow e_3]] \Rightarrow [r_3(e_1) \Rightarrow e_3]$
$(h, r_1, t) \Leftrightarrow (t, r_2, h)$	$[[r_1(h) \Rightarrow t] \Rightarrow [r_2(t) \Rightarrow h]] \wedge [[r_2(t) \Rightarrow h] \Rightarrow [r_1(h) \Rightarrow t]]$

Table 5. FOLs and their corresponding mathematical expressions.

FOL	Mathematical Expression
$r(h)$	$\mathbf{r} + \mathbf{h}$
$a \Rightarrow b$	$\mathbf{a} - \mathbf{b}$
$h \in C$	$\mathbf{h} \cdot \mathbf{C}$ ($\mathbf{C}$ is a matrix)
$a \wedge b$	$\mathbf{a} \otimes \mathbf{b}$
$a \Leftrightarrow b$	$(\mathbf{a} - \mathbf{b}) \otimes (\mathbf{a} - \mathbf{b})$

Complex embeddings for logical rules. Aiming to tackle the transitivity and antisymmetry of logical rules, Wang et al. [68] introduced TARE. In TARE, The fundamental concept revolves around acquiring KG embeddings by incorporating triples, existing relations, and logical rules. Moreover, the transitivity and antisymmetry properties of logical rules are exploited to estimate the relative ranking of relation types within the logical rules. Logical rules are directly incorporated into the embedding of relation types, instead of instantiating them with specific entity instances. As a result, the learned embeddings are not only compatible with triples, but also compatible with logical rules, and the embeddings of relation types are approximately ordered.

Complex-NNE-AER [70] also simultaneously models the real and imaginary components of entity and relation embeddings. This approach integrates non-negative (NNE) constraints on entity embeddings and approximate entailment constraints on relations (AER) into the embedding model. The objective is to incorporate prior beliefs into the structural characteristics of the embedding space while preserving the quality of embeddings and their transferability. NNE contributes to acquiring concise and explainable entity embeddings, whereas AER further encodes the logical rules that underlie relations into their embeddings. Complex-NNE-AER is notable for its simplicity, effectiveness, broad applicability, and interpretability. It enriches the structural properties of the embedding space, rendering it well-suited for large-scale KGs.

Joint embedding with relational paths. RPJE [72] integrates rule learning, embedding, and path searching. Figure 9 illustrates the model architecture of RPJE. RPJE improves the generalization ability of KGE and the supplementary semantic structure provided by paths. Consequently, it effectively mitigates the challenges encountered by embedding models in adapting to sparse KGs and enhancing interpretability.

Prior logical rule injection. To address the stability concern associated with rule learning, Rule [73] leverages embeddings to model logical rules and triples. By unifying the embeddings of entities, relations, and logical rules within a shared space, Rule learns the embedding for each logical rule. To execute logical rule reasoning, this method adopts a “soft approach” and calculates confidence scores for individual basic rules. In contrast to conventional embedding techniques, Rule allows the incorporation of prior logical rule information into the embedding space, which in turn enhances the generalization capability of KGE. Moreover, the confidence scores assigned to the learned rules contribute to the improvement of the logical rule reasoning process by softly controlling the influence of each rule. As a result, Rule effectively mitigates the issue of poor robustness commonly observed in rule learning.

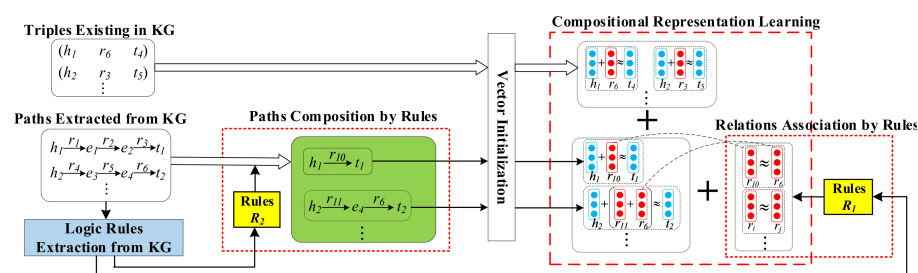


Figure 9. Overall architecture of RPJE. Firstly, Horn rules of varying lengths are extracted from the KG and encoded for embedding. Subsequently, RPJE meticulously combines paths using rules of length 2, establishes semantic associations between relations using rules of length 1, and effectively constrains relation embeddings. The optimization process takes into account the confidence of each rule to ensure the reliability of rule application in embedding representation.

Open Challenges. Rule embedding-based KGR has been proven to enhance the efficiency and accuracy of KGR. Nevertheless, it should be noted that these methods fundamentally rely on embedding representation to capture the structure of the KG, which leads to an inherent lack of interpretability in the overall model.

5.2. Mutually Enhanced between Embedding and Rule for KGR

Rule learning and embedding learning exhibit mutual complementarity. On one hand, rules contribute supplementary triples for reasoning in sparse KGs, thereby enhancing the accuracy of embedding learning. On the other hand, embeddings that include abundant semantic encoding facilitate the conversion of rule learning from discrete graph search to vector space computations, thereby notably shrinking the search space. Consequently, approaches that mutually enhance embedding between embeddings and rules can strike an equilibrium between efficiency enhancement and interpretability in the process of reasoning.

Iterative updating rules and embeddings for sparse KGR. IterE [75] adopts an approach to update both rules and embeddings, aiming to facilitate explainable reasoning in sparse KGs. In each iteration, IterE infers new rules based on the updated embeddings. The IterE algorithm consists of three key components: (1) embedding learning, (2) axiom induction, and (3) axiom injection. The model framework and detailed reasoning steps of IterE is shown in Figure 10. IterE improves the quality of sparse embeddings through the utilization of axioms, enhances the efficiency and effectiveness of rule learning through embeddings, and the iterative training ensures superior performance in link prediction and yields high-quality rule learning.

The challenge of learning high-quality rules solely based on the information within KGs arises due to the presence of missing facts. To address this challenge, RuLES [74] incorporates embedding model feedback to learn high-quality rules in the context of missing triples. The framework of RuLES, as shown in Figure 11, consists of three modules: rule learning, embedding learning, and rule evaluation. By integrating the evaluation of rule quality in sparse KGs with the feedback from the embedding module, RuLES effectively learns higher-quality rules for reasoning purposes.

Combination of embeddings and rules for KGR with multi-type rules/predicates. To enhance the accuracy of KGR in large-scale KGs with numerous predicates, RLvLR [76] employs a combination of embedding learning and sampling strategies to acquire rules. RLvLR primarily introduces three techniques: (1) subgraph sampling, which diminishes the size of the input KG by eliminating triples that have low relevance to the target predicate; (2) parameter embedding, which evaluates the quality of rules; (3) rule quality evaluation based on matrix operations, which mitigates the need for computationally expensive query response operations. RLvLR demonstrates notable performance in terms of rule-learning quality and efficiency, surpassing similar methods in link prediction accuracy.

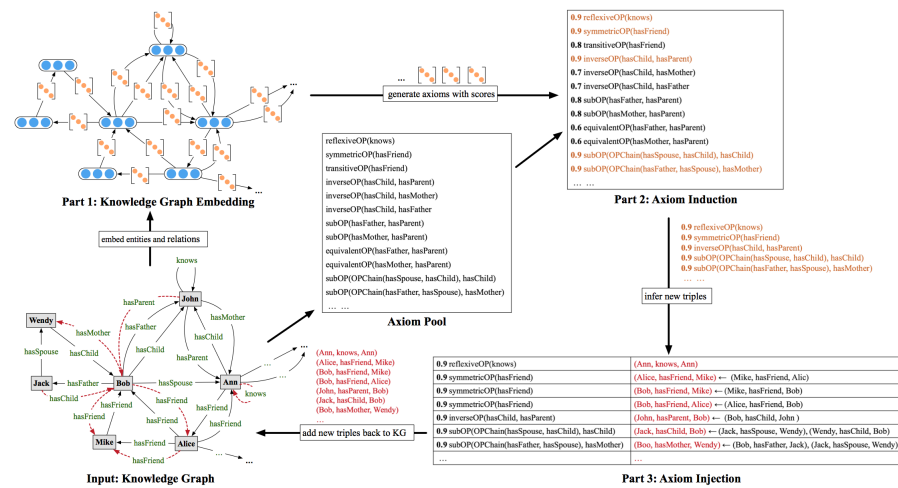


Figure 10. A detailed example of the process of IterE. (1) *Embedding learning module* involves acquiring embeddings by utilizing existing triples in the KG and inferred triples from rules. (2) *Axiom induction module* generates an axiom pool through a pruning strategy and subsequently calculates relation embeddings based on the rule conclusions, assigning scores to each axiom in the pool. (3) *Axiom injection module* leverages axioms to deduce new triples concerning sparse entities based on fundamental axioms, which are then injected into the KG to enhance the embeddings of sparse entities.

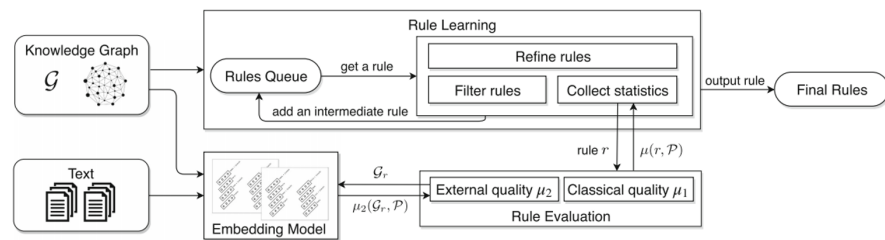


Figure 11. Detailed Steps of RuLES. In the rule-learning module, rules are constructed based on the KG $\mathcal{G}$. The rule evaluation module employs $\mathcal{G}$ and quality scoring functions f provided by the embedding model to assign quality score μ , including rule description quality score μ_1 and embedding-based prediction probability score μ_2 . The embedding module learns the embeddings of triples by utilizing both the KG and text corpus.

In the context of large-scale KGs, the handling of numerous basic rules poses a challenge for methods like IterE and RuLES, thereby affecting the transferability and efficiency of reasoning. To tackle this issue, UniKER [77] capitalizes on the embedded knowledge within rules to facilitate mutual enhancement between rule learning and embedding in an efficient manner. In the pursuit of rule determination, UniKER devises an enhanced forward chaining iterative algorithm that incorporates “lazy inference” to avoid the use of many inactive base predicates and rules. Subsequently, it updates the system using embedding-based variational inference. This approach utilizes embedding techniques to add valuable hidden triples and remove incorrect triples from both the KG and the resulting reasoning outcomes. UniKER effectively harnesses the knowledge encapsulated within rules to acquire superior embeddings, while the embeddings, in turn, fortify forward chaining, rule learning, and embeddings by offering triples.

Open challenges. Approaches that unify embedding techniques and logical rules strive to expedite the acquisition of high-quality rules while augmenting the interpretability of the reasoning process and results, with a particular focus on link prediction and rule-learning domains. Nonetheless, as the number of reasoning hops escalates, the performance of reasoning becomes increasingly susceptible to the sparsity of the KG, thereby resulting in a decline in reasoning efficiency.

6. Jointly Using Neural Networks and Logical Rules for KGR

Neural networks have demonstrated outstanding performance in diverse domains, including computer vision, natural language processing, and network link prediction. NNs can effectively capture implicit features and correlations in data, showcasing superior capabilities in terms of speed, accuracy, robustness, and generalization.

However, NNs are limited in their ability to provide explainable reasoning results. Consequently, the integration of NNs and rules has emerged as a prominent research focus in KGR. This integration aims to achieve efficient and explainable reasoning. Depending on the dominant role played by either NNs or rules in the reasoning process, these methods can be categorized as rule-enhanced NN modeling methods and NN-enhanced rule-learning methods. Table 6 summarizes the relevant information on these methods.

6.1. Rule-Enhanced NN Modeling for KGR

The primary objective of rule-enhanced NN modeling in KGR is to exploit logical rules for augmenting the embedding and prediction capabilities of NNs. In this approach, NNs are trained to learn embeddings not only for the entities and relations observed in the KG but also for the inferred triples or basic rules derived from predefined logical rules. By incorporating these rules, the training process of the NN is guided, while the inclusion of the inferred triples enriches the training dataset. Consequently, this integration leads to reasoning results that are more explainable and it enhances the efficiency. These types of methods can be further categorized into approaches based on recurrent neural networks (RNNs) [80] and GNNs.

6.1.1. RNN-Based Approaches

RNN leverages its advantages in handling sequential information, modeling long-term dependencies, parameter sharing, and providing interpretability to accomplish KGR tasks.

Logic attention network (LAN) [81] aggregates neighbor embeddings using attention weights based on both rules and networks. LAN is built upon the encoder-decoder framework using RNN and combines neighbor attention through a combination of logical rule mechanisms and NN mechanisms. The logical rule mechanism is used to estimate weights at a coarse-grained relation level, quantifying the confidence $\mathcal{P}(r \Rightarrow q)$ of the potential dependency between rules, where q denotes the query relation. If r is an important neighbor of q , it should have a sufficiently large $\mathcal{P}(r \Rightarrow q)$ and a smaller $\mathcal{P}(r' \Rightarrow r)$, where r' denotes other relations excluding r and q . Subsequently, logical attention weights are defined as follows:

$$\alpha_{j|i,q}^{\text{Logic}} = \frac{\mathcal{P}(r \Rightarrow q)}{\max(\{\mathcal{P}(r' \Rightarrow r) | r' \in N_{\mathcal{R}}(e_i) \wedge r' \neq r\})}. \quad (12)$$

Table 6. Summary and comparison of KGR methods that jointly use NNs and logical rules.

Category	Subclass	Method	Year	Model Interpretability	Unseen Entities	Sparse KGR	Complex Rule Form
Rule-Enhanced NN Modeling	RNN-based	LAN [81]	2019	×	×	×	✓
		GRAFT-Nets [82]	2019	×	×	✓	✓
		RuleGuider [83]	2019	×	×	×	✓
		SparKGR [84]	2022	×	×	✓	✓
	GNN-based	pLogicNet [85]	2019	×	×	×	✓
		VN [86]	2020	×	×	×	✓
		PGAT [87]	2020	×	×	×	✓
		ExpressGNN [88]	2020	×	×	×	✓

Table 6. Cont.

Category	Subclass	Method	Year	Model Interpretability	Unseen Entities	Sparse KGR	Complex Rule Form
NN-enhanced rule learning	Neural Theorem Proving	NTP [89]	2017	✓	×	✓	×
		GNTP [90]	2019	✓	×	✓	×
		CTP [91]	2020	✓	×	✓	×
	Matrix Multiplication	TensorLog [92]	2016	✓	×	×	×
		Neural LP [93]	2017	✓	×	×	×
		DRUM [94]	2019	✓	✓	×	×
		Num-Neural LP [95]	2020	✓	×	×	×
	Neural Inductive Learning	NLM [96]	2019	✓	×	×	✓
		NLIL [97]	2020	✓	×	×	✓
		GraIL [98]	2020	✓	✓	×	✓
		GCR [99]	2021	✓	✓	✓	✓
		CoMPILE [100]	2021	✓	✓	×	✓
		TACT [101]	2021	✓	✓	×	✓
		RPC-IR [102]	2021	✓	✓	×	✓
		ConGLR [103]	2022	✓	✓	×	✓
		SNRI [104]	2022	✓	✓	×	✓
		BERTRL [105]	2022	✓	✓	×	✓
		CBGNN [106]	2022	✓	✓	×	✓
		RNNLogic [107]	2021	✓	×	×	✓
		CogKR [108]	2020	✓	✓	✓	✓
		CSR [109]	2022	✓	×	✓	✓

By computing $\alpha_{j|i,q}^{\text{Logic}}$, the issue of perceiving query relation and neighborhood redundancy perception strength is addressed. NN mechanism represents neighbor weights in a more granular manner, enabling it to calculate the importance weights $\alpha_{j|i,q}^{\text{NN}}$ after embedding the neighbors. Ultimately, LAN combines $\alpha_{j|i,q}^{\text{Logic}}$ and $\alpha_{j|i,q}^{\text{NN}}$ by weighting and summing them to effectively describe the importance of neighbor entities. LAN demonstrates enhanced representation learning capability for entities, resulting in notable enhancements in reasoning performance.

GRAFT-Nets [82] addresses the challenge of reasoning on incomplete KGs augmented with text rule information. This method offers two significant improvements. Firstly, it introduces heterogeneous update rules for both KG nodes and text nodes. By leveraging LSTM [110], the propagation information between text rule nodes is effectively updated. Secondly, it draws inspiration from the personalized PageRank (PPR) [111] and proposes a directed propagation approach. PPR ensures that graph embeddings consistently follow the path direction originating from the seed node of the question. GRAFT-Nets excels at extracting answers from specific question subgraphs that include text rule information, entities, and relations.

In some studies, LSTM is also employed as a policy encoder in the context of DRL, where rules are utilized to provide guidance for the policy learning of the agent.

RuleGuider [83] initially generates high-quality rules using AnyBURL and employs them to provide reward supervision for the agent. Subsequently, entity agent and relation agent are set for exploration. The entity agents generate entity distributions pertaining to the selectable entities, while the relation agents sample relations and prune the entity space based on the selected relation. Following this, the entity agents further sample entities from the pruned entity space. Finally, the hit reward is computed based on the selected entities, and the rule-guided reward is acquired from a pre-mined set of rules by considering the relation path. These rewards are then utilized to optimize the agent policy (Figure 12).

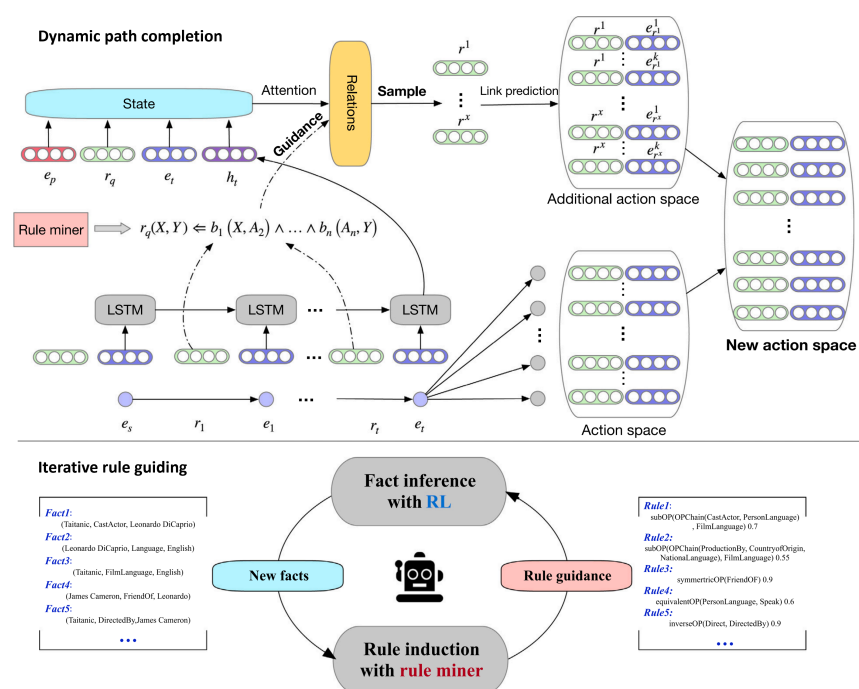


Figure 12. Within SparkGR, the agent utilizes two strategies for selecting the next action: dynamic path completion and iterative rule guidance. The probabilities derived from these strategies are then weighted and aggregated to determine the agent's subsequent action.

SparkGR [84] is a multi-hop reasoning method developed specifically for sparse KGs, which employs rule-guided dynamic reasoning to address missing paths, thereby expanding the action space available to the DRL agent and mitigating the issue of sparsity. SparkGR incorporates iterative optimization techniques for rule induction and fact reasoning, allowing the integration of global KG information to guide the exploration process. This approach effectively reduces sparsity, enhances path exploration efficiency, and improves reasoning accuracy while maintaining interpretability.

6.1.2. GNN-Based Approaches

MLNs have the ability to leverage FOL domain knowledge and handle uncertainty, but they struggle to adapt to complex graph structures. GNNs have high reasoning efficiency but lack the ability to utilize domain knowledge. GNN-based approaches utilize the combination of GNNs and MLNs for KGR. GNNs are employed to aggregate neighboring information on the KG and perform multi-hop reasoning, while MLNs are used for logical rule-based reasoning.

The pLogicNet [85] defines the joint distribution of all possible triples using MLNs combined with FOL. It then efficiently optimizes this distribution through the variational EM algorithm. During the E-step, the variational distribution is parameterized using a KGE model, and amortized mean-field [112] inference is employed to compute the likelihood probability of unobserved triples. In the M-step, pseudo-likelihood probabilities are calculated based on both observed and predicted triples to update the weights of logical rules. Ultimately, training is conducted using gradient descent.

Most embedding models assume the availability of all test entities during training, resulting in the need for time-consuming retraining of embeddings. To tackle this issue, VN network [86] employs a graph convolutional network (GCN) as a solution. Initially, a rule-based reasoning approach is introduced to mitigate the sparsity of neighbors by assigning soft labels to neighbors through constraint rules. Furthermore, GCN is employed to simultaneously learn logical and symmetric path rules, enabling the capture of intricate patterns. Additionally, an iterative learning technique is adopted to capture the interaction

between embeddings and virtual neighbor predictions, facilitating the updating of rule learning. The framework of VN is shown in Figure 13. VN network not only achieves superior reasoning accuracy but also exhibits significantly enhanced robustness against neighbor sparsity.

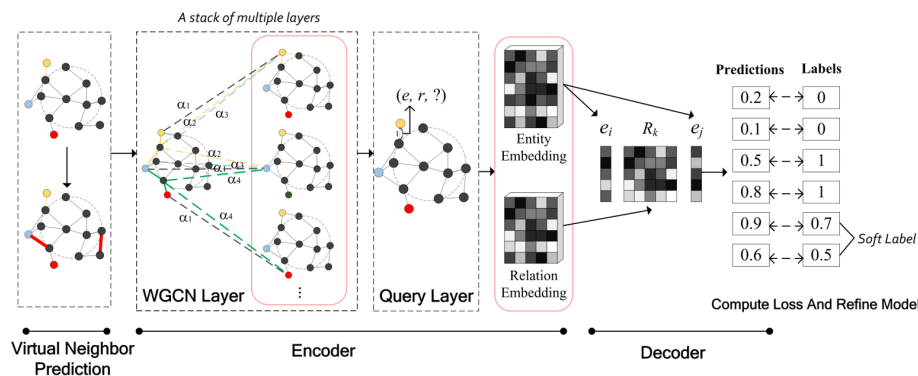


Figure 13. Overall architecture of VN, primarily comprises three components: (1) rule-based virtual neighbor prediction; (2) an encoder that captures structural information and embeds entities using weighted graph convolutional network (WGCN) [113]; (3) a decoder that computes edge probabilities and refines the model using soft labels.

PGAT [87] is a probabilistic logic graph attention network that integrates MLN and graph attention network (GAT). By employing the variational EM algorithm, PGAT effectively combines FOL and GATs to optimize the joint distribution of all possible triples defined by MLN. The framework of the PGAT method is shown in Figure 14.

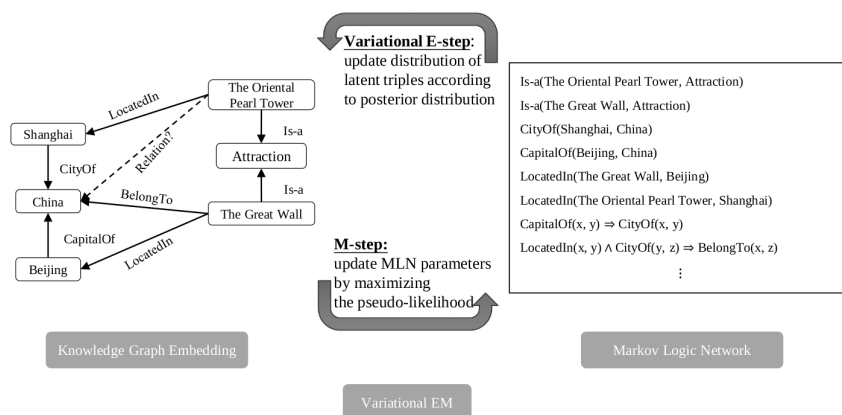


Figure 14. Detailed step of PGAT. During the E-step, unobserved triplets are inferred using GAT embeddings. In the M-step, the weights of MLN rules are updated based on both observed triples and triples inferred from GAT embeddings.

ExpressGNN [88] expands the scope of MLN reasoning to address large-scale KGR challenges through the integration of prior knowledge from logical rules of MLN and the structural knowledge captured by GNN in KGs. As shown in Figure 15, ExpressGNN initially models the joint probability distribution of observed and latent variables using MLN, optimizing it through the evidence lower bound of log-likelihood. ExpressGNN not only yields enhanced reasoning accuracy but also allows for a trade-off between model compactness and expressiveness by adjusting the dimensions of GNN and embedding components. Additionally, ExpressGNN effectively tackles few-shot and zero-shot learning problems involving a limited number of target predicates or instances lacking labels.

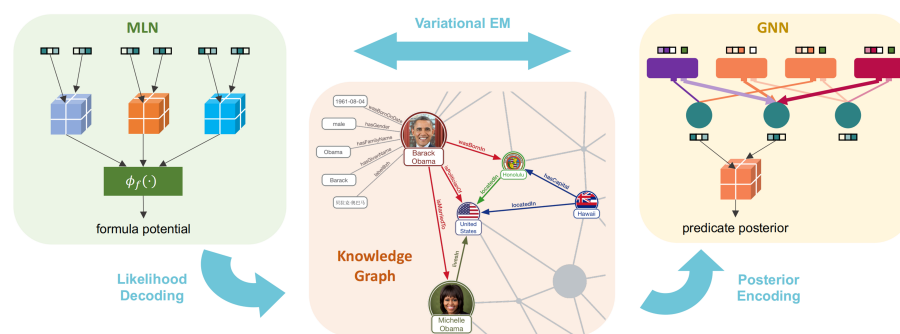


Figure 15. Overall architecture of ExpressGNN. During the E-step, an approximate posterior distribution is derived via mean-field approximation, with GNN employed to parameterize the variational posterior. In the M-step, only the base predicates are retained to compute the pseudo-likelihood and refine the weights of logical formulas in the variational posterior of MLN.

6.1.3. Open Challenges

Incorporating rules to enhance NN modeling offers support in tasks like training and parameter learning, leading to explainable reasoning results. Nonetheless, these methods do not uncover underlying patterns within the internal structure of models and parameters, thus failing to fundamentally address the inherent “black-box” nature of NNs. Conversely, approaches that combine rules with DRL provide enhanced interpretability, albeit relying on the exploration of paths rather than mining the interpretability of the NNs.

6.2. NN-Enhanced Rule Learning for KGR

NN-enhanced rule-learning methods exploit the capabilities of NNs to address the challenges posed by uncertain and fuzzy data, thereby narrowing down the search space for rule reasoning and facilitating the discovery of high-quality rules for KGR. These methods include various approaches, such as neural theorem proving, matrix multiplication, and neural inductive learning.

6.2.1. Neural Theorem Proving Approaches

The neural theorem proving (NTP) [89] approaches are built upon the Prolog programming language [114]. Prolog involves steps such as predicate matching, rule resolution, and backtracking. By combining recursive matching algorithms with NN embeddings, this approach performs knowledge reasoning and offers explainable paths for theorem proving.

NTP recursively constructs NNs through the utilization of backward chaining algorithm. It operates on dense vector representations of symbols and employs a differentiable prover (DP) to enable differentiable proof throughout the entire process of KGR. NTP takes atoms, rules, and their associated proof states as input and produces a fresh list of proof states as output. In the initial phase, three DP modules are employed to verify queries on the KG. Subsequently, proof aggregation is performed to evaluate the success score of the target. Finally, NTP is utilized for ILP by training NNs using the gradient descent algorithm to acquire explainable rules from data.

NTP is capable of inferring facts from incomplete KGs and introduces explainable FOL rules. Nevertheless, the enumeration and evaluation of all bounded-depth proof paths for a given target in the execution process of this method pose efficiency challenges when applied to large-scale KGs.

GNTN [90] aims to tackle the above challenges. GNTN integrates the KG and text reasoning by embedding logical facts and natural language sentences in a shared embedding space. To overcome the limitations of NTP, GNTN employs three techniques. Firstly, it utilizes fact embeddings to select the closest neighbor facts for proving sub-goals, and relation embeddings to choose rules for expansion. As a result, the number of candidate proof paths is reduced, diminishing the complexity of the space. Secondly, GNTN introduces

an attention mechanism for rule induction to handle known predicates, which improves the efficiency of the model and reduces the time complexity. Lastly, GNTP extends NTP to natural language by incorporating an end-to-end differentiable reading module. This enables the joint embedding of predicates and text patterns in the shared space, thereby leveraging textual information for reasoning. Notably, GNTP maintains the interpretability of the NTP approach while alleviating the complexity and scalability limitations.

During the training process, NTP encounters difficulties in handling a large number of rules or inference steps, as it necessitates considering all the rules that explain a given goal or sub-goal.

Aiming at the above setback, CTP [91] leverages an NN to dynamically generate a minimal rule set and employs gradient-based optimization to learn an optimal rule selection strategy. For each sub-goal, the selection module dynamically generates a minimal rule set and presents three rule selection methods: neural goal reformulation (NGR), attentive goal reformulation (AGR), and memory-based goal reformulation (MGR). NGR defines the selection operation as a linear function of the target predicate, to reduce the set of rules that are conditioned on the sub-goal for end-to-end training in downstream inference tasks. AGR introduces an attention distribution over the relations set based on the predicate to incorporate valuable priors into the selection module. MGR stores rules in a differentiable memory and generates an attention distribution over the rules given the goal of key-value memory retrieval. CTP demonstrates superior transferability compared with NTP.

Limitations. The computational complexity of NTP methods is often high due to the intricate nature of the matching, solving, and backtracking procedures involved. Moreover, the sequential pattern of inferring before proving results in an excessive complexity of the model steps. Lastly, the requirement to satisfy multiple assumptions for differentiability in theorem proving may introduce biases.

6.2.2. Matrix Multiplication-Based KGR

This methodology defines relations as two-dimensional matrices and employs matrix multiplication to model multi-hop reasoning. RNNs are employed to simulate rule-based reasoning. The differentiability of matrix multiplication facilitates synchronized learning of rule parameters and structures, thus enabling the effective handling of intricate KGR tasks. Furthermore, the reasoning process is accomplished hierarchically through matrix multiplication at each layer, thereby furnishing a more lucid explainable foundation for the results.

TensorLog [92] is a probabilistic deductive framework, which can infer weighted chain logical rules for knowledge reasoning and explanation. In TensorLog, each entity in the KG is represented using one-hot embeddings, and each relation R is represented by a matrix $\mathbf{M}_R$, where $\mathbf{M}_R(i, j) = 1$ if $R(i, j)$ exists in the KG. Then, given a rule $\gamma : R(X, Y) \leftarrow P(X, Z) \wedge Q(Z, Y)$ and a head entity X , logical inference of the answer can be performed by executing matrix multiplication $\mathbf{M}_P \cdot \mathbf{M}_Q \cdot \mathbf{v}_X$. Since query relations can be explained by multiple rules, the score of the query relation is calculated by combining all rules:

$$\sum_{\gamma} \alpha_{\gamma} \prod_{k \in \beta_{\gamma}} \mathbf{M}_{R_k}, \quad (13)$$

where γ denotes all possible rules, α_{γ} is the confidence level, and β_{γ} is an ordered list of all predicates in γ . In the reasoning process, given a head entity X , the probability of each correct answer Y satisfying $R(X, Y)$ is maximized:

$$\max_{\{\alpha_{\gamma}, \beta_{\gamma}\}} \sum_{X, Y} \mathbf{v}_Y^T \left(\sum_{\gamma} \alpha_{\gamma} \prod_{k \in \beta_{\gamma}} \mathbf{M}_{R_k} \mathbf{v}_X \right). \quad (14)$$

More specifically, the transformation of each clause in the logical rule involves mapping it to a *factor graph*. In this graph, nodes represent variables within the rule, while edges

represent predicates or relations. Subsequently, belief propagation is employed for diverse query types on the factor graph, and the necessary steps for message passing are expanded into a differentiable function to facilitate the computation of probabilities. Lastly, these differentiable functions are recursively integrated to carry out reasoning in a KG including multiple interconnected rules and predicates.

TensorLog lacks the ability to generate novel logical rules, and the association of each rule with a parameter leads to a high computational complexity when enumerating rules. To overcome this limitation, neural LP [93] is proposed as an enhancement. In neural LP, a swap of the summation and multiplication in Equation (14) takes place, and the weights of the rules are decomposed into weights of the predicates present within the rules. Additionally, to handle rules of varying lengths, neural LP introduces a recursive methodology for dynamically modeling rules:

$$\begin{aligned} \mathbf{u}_0 &= \mathbf{v}_X, \\ \mathbf{u}_t &= \sum_k |\mathbf{R}| a_t^k \mathbf{M}_{R_k} \left(\sum_{\tau=0}^{t-1} b_t^\tau \mathbf{u}_\tau \right) \quad \text{for } 1 \leq t \leq T, \\ \mathbf{u}_{T+1} &= \sum_{\tau=0}^T b_{T+1}^\tau \mathbf{u}_\tau, \end{aligned} \quad (15)$$

T denotes the maximum length of rules, $|\mathbf{R}|$ denotes the number of predicates within the KG, and $\mathbf{u}$ serves as an auxiliary storage vector for the input entity $\mathbf{v}_X$. Within each recursion step, the previous auxiliary storage vectors are averaged using a memory attention vector denoted as $\mathbf{b}_t$. TensorLog operations are then executed by leveraging an operation attention vector denoted as $\mathbf{a}_t$. Ultimately, the average weights of all auxiliary storage vectors are calculated, and the length of rules is controlled through the utilization of attention mechanisms. To model and solve for $\mathbf{a}_t$ and $\mathbf{b}_t$, neural LP employs an RNN architecture. By utilizing the computed attention vectors, the reconstruction of rules and their associated confidences is facilitated.

The capabilities of neural LP are constrained when it comes to handling numerical features. Extending upon the foundations of neural LP, Num-Neural LP [95] incorporates dynamic programming and cumulative sum operations to articulate numerical comparison operators, and combines them with sparse operators to accommodate dense computations.

The end-to-end differentiable model DRUM [94] is designed to extract variable-length FOL rules from a given KG. DRUM simultaneously learns the structure of rules and confidence scores, enabling inductive and explainable reasoning over KGs. The process begins by transforming rule inference into a sequence of multiplicative computations between vectors. To facilitate the learning of variable-length rules, a specialized relation is introduced, with an identity matrix serving as its adjacency matrix. Furthermore, a confidence tensor is incorporated to prevent the learning of inaccurate rules with excessively high confidence. Lastly, a bidirectional RNN is harnessed to capture the sequential information of atoms within rules, thereby modeling the relationships between the rule heads and bodies. DRUM supports inductive reasoning and effectively handles previously unseen entities, thereby showcasing its remarkable interpretability.

Limitations. Constrained by the inherent properties of matrix multiplication, the aforementioned approaches are limited to learning sequential logical rules and are unable to accommodate intricate rule forms, such as tree structures and conjunctions. Moreover, the reliance of these methods on specific head entities for rule inference may adversely affect their capacity for generalization. The transformation process from entities, relations, and rules to matrices hinges on embedding methods, thereby diminishing the overall robustness of the model.

6.2.3. Neural Inductive Learning for KGR

Neural inductive learning methods leverage GNNs and RNNs to learn inductive logical rules and facilitate reasoning and prediction. GNNs, utilizing graph-structured data

as their representation scheme, enable efficient computations on nodes and relations. They incorporate crucial information from hidden layers, neighbor nodes, and the overall graph structure, thereby making them highly suitable for rule learning within KGs. RNNs excel in capturing sequential information and can also be effectively employed in KGR.

Traditional NN-based approaches. NLIL [97] addresses non-chain rules by employing the combination of primitive statements (PSs). A PS represents a predicate that is applicable to logical variables or specific operator outcomes. A unary PS entails a relational path originating from a variable, while binary PS consists of two distinct variable-based relational paths. PS can also be logically merged using operators, enabling the representation of tree-like and conjunction logical rules, as shown in Figure 16. To ascertain the significance weights of various PS logical combinations in rule learning, three stacked Transformers [115] are employed to acquire separate attention vectors.

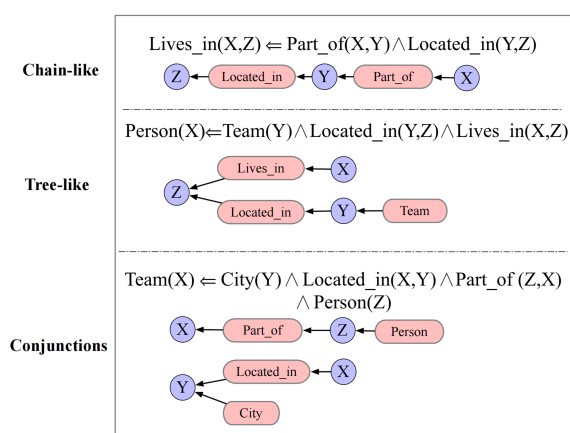


Figure 16. Three types of rules that NLIL can learn.

NLM [96] leverages NNs as function approximators and integrates logic programming to handle objects including properties, relations, logical connectives, and quantifiers. NLM assigns fixed predicates as either True or False on a predetermined set of objects, representing logical predicates using tensors. The neural architecture for rule learning is then established using a multi-layer perceptron. Following training on small-scale tasks, NLM exhibits the capability to extend and generalize the acquired rules to larger-scale tasks.

GNN-based approaches. GCR [99] employs adjacent link information to perform relation reasoning on KGs. GCR makes use of GNNs to leverage the message-passing capability and aggregate valuable information from neighboring links. Initially, the graph structure is transformed into logical expressions of Horn rules, converting the KGR task into a neural logic reasoning problem. Subsequently, each triple is encoded as a predicate embedding, while each relation is modeled as a neural module for triple encoding. Ultimately, by utilizing the encoded predicate embeddings, the neural modules construct the network structure based on the modeled Horn rules, and the model parameters are learned through backpropagation. GCR excels at effectively utilizing the logical relations between links, eliminating the need for manual predefined rules. It also demonstrates significant interpretability and transferability, making it particularly well-suited for sparse KGR.

GraIL [98] discovers FOL rules by learning subgraph embedding representations and leveraging the subgraph structure surrounding candidate relations to make relation predictions. Figure 17 illustrates the three main steps of GraIL during relation prediction. GraIL demonstrates the capability to acquire a valuable subset of FOL rules and perform entity reasoning beyond the scope of the training set.

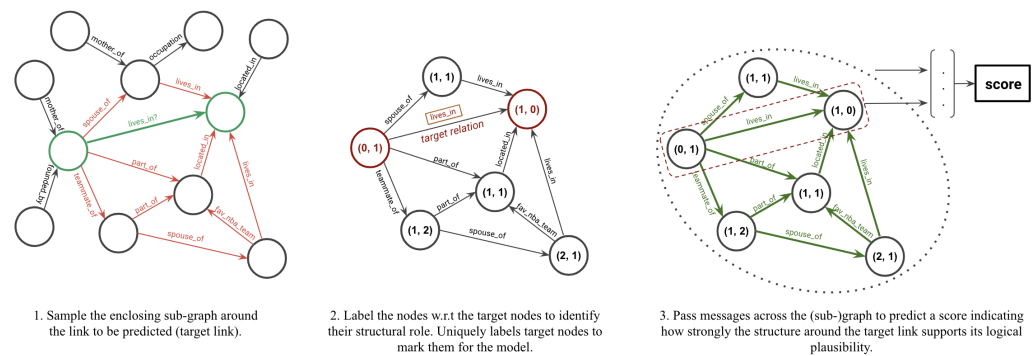


Figure 17. Visual illustration of GraIL for inductive relation prediction. (1) *Subgraph extraction*: Extract closed subgraphs around the target node, mine logical evidence containing the relationships required for reasoning between the inferred target nodes, and explicitly encode rules. (2) *Node labeling*: Utilize the distance between nodes and surrounding subgraph nodes to construct a node feature matrix as input for GNN, initializing the neural message passing algorithm. (3) *GNN scoring*: Employ GNNs to score the likelihood of triples by iteratively updating node representations through the combination of aggregating node and neighbor representations.

The omission of directional properties in the extraction of closed subgraphs for target triples renders GraIL incapable of handling asymmetric and antisymmetric relations. To address this limitation, CoMPILE [100] introduces a node-edge communication message-passing mechanism as a substitute for the original GNN to assess the significance of relations. CoMPILE initially extracts directed closed subgraphs for triples and subsequently extends the communication message-passing network framework to promote the interaction of information between entities and relations, facilitating the concurrent update of edge and entity embeddings. Moreover, an edge-aware attention mechanism is employed to aggregate local neighborhood features and gather comprehensive entity information, thereby enriching entity and relation representations.

In response to the lack of consideration for semantic correlations in GraIL, TACT [101] incorporates correlations between relations in subgraphs and encodes the relation-related network to improve the encoding of closed subgraphs, thus enabling topology-aware correction.

To strike a balance between entity-independent relation modeling and the interpretability of discrete logical reasoning, ConGLR [103] constructs a contextual graph that incorporates relation paths, relations, and entities based on the extraction of closed subgraphs for target head and tail entities. The subgraph and contextual graph are processed using a GCN with perceptual attention. The relation paths are used as rule bodies, and the target relations serve as rule heads, combining neural computation and logical reasoning to compute confidence scores.

To address the challenge of handling sparse subgraphs, SNRI [104] captures complete adjacency relations of entities in subgraphs by leveraging neighboring relation features and paths. It globally models neighboring relation paths through mutual information maximization, incorporating comprehensive relation information into closed subgraphs and modeling adjacent relation paths to enhance the performance of inductive reasoning predictions.

BERTRL [105] integrates pre-trained language models with relational learning. Initially, the KG is linearized and used as input for BERT [116]. BERT then encodes the inference paths. Finally, the scores of the triple paths and inference paths are aggregated. The inference steps of BERTRL are shown in Figure 18.

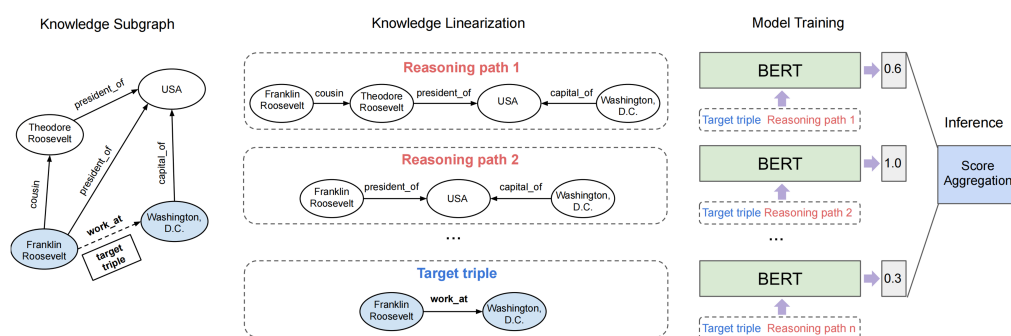


Figure 18. The BERTRL pipeline.

To acquire entity-specific semantic relations from latent rules and tackle the issue of inadequate supervision caused by sparse rules in subgraphs, RPC-IR [102] adopts a relation pair comparison strategy utilizing inductive paths. RPC-IR includes three distinct stages: (1) extraction of paths from closed subgraphs associated with target triples and generation of positive and negative samples for relation paths to facilitate comparisons; (2) utilization of GCNs to derive embeddings for positive and negative samples; (3) evaluation of the target triple based on the subgraph and relation paths, employing a unified training approach that concurrently considers both supervision and comparison information. By implementing these steps, rules are ascertained based on the confidence scores associated with relation paths.

The aforementioned approach regards rules as paths, and the task of mining paths in large-scale KGs often encounters a substantial search space. To tackle this challenge, CBGNN [106] adopts an algebraic topology perspective by considering logical rules as cycles. It introduces a GNN that operates within the cyclic space, exploiting its message-passing capability to conduct implicit algebraic operations and acquire representations of cycles. Subsequently, the model learns rules by exploring the linear structure within the cyclic space, thereby enabling KGR. Figure 19 provides an illustration of the method. Compared to methods based on rule paths, the CBGNN exhibits enhanced efficiency and offers valuable insights for integrating advanced topological information into graph representation learning.

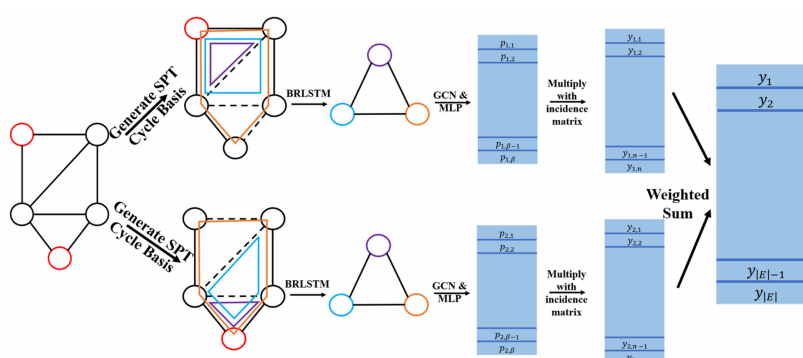


Figure 19. Two stages of CBGNN. In the first stage, cyclic bases are constructed, and a cyclic graph is created for each cyclic base. The graph information can be transformed into node features within a novel graph, facilitating the second stage. In the second stage, a GNN is built upon the cyclic graph to learn the confidence values associated with cycles and map them to confidence values for target triples.

RNN-based approach. To address the challenges of a large search space in neural logic programming methods and the poor efficiency caused by sparse rewards in DRL, RNNLogic [107] treats logical rules as latent variables and trains rule generators and reasoning predictors accordingly. The rule generator employs RNN parameterization.

During the iterative optimization process, the reasoning predictor is initially updated to explore logical reasoning rules. Subsequently, the EM algorithm is applied, where in the E-step, a subset of high-quality rules is selected through posterior inference from the generated rules, and in the M-step, the rule generator is updated using the selected rules. The model framework of RNNLogic is illustrated in Figure 20.

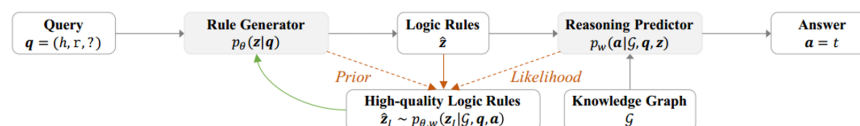


Figure 20. Reasoning steps of RNNLogic. Here, q denotes the query, z denotes the latent representation of rules, θ and w denote the model parameters. $p_{\theta}(z|q)$ denotes the prior probability of generating the rule latent distribution from the query, and $p_w(a|G, q, z)$ denotes the likelihood probability of determining the answer based on the KG, rule latent representation, and query.

Reasoning in scenarios with scarce samples. To tackle the challenge of few-shot KGR, CSR [109] adopts a transformative approach by reframing the task as an inductive reasoning problem. CSR proposes an innovative encoder-decoder architecture, leveraging GNNs to effectively capture shared subgraphs between support and query triples. Furthermore, a self-supervised pre-training scheme is devised to reconstruct automatically sampled subgraphs. Notably, this method obviates the need for a separate pre-training phase, allowing direct prediction of the desired few-shot task.

Regarding one-shot KGR, CogKR [108] introduces two distinct modules: summarization and inference. The summarization module aims to capture the fundamental relations within a given instance, while the inference module leverages these summarized relations for reasoning purposes. In the summarization module, entity pair embeddings are generated using GNNs to effectively encode the relations between entity pairs. As for the inference module, two systems are established to construct a cognitive graph that serves as a repository for storing retrieved information and reasoning results. The structural properties of the cognitive graph enable the model to consolidate evidence from various reasoning paths and offer graphical explanations of the reasoning process. Lastly, a DRL approach is employed to translate the modeling of graph structures into a policy optimization problem.

6.2.4. Open Challenges

Overall, NN-enhanced rule-learning approaches demonstrate high efficiency and accuracy, as well as strong interpretability. However, the utilization of NNs for rule learning introduces a black-box nature that hinders the understanding of the learning process and decision-making. Moreover, data biases present in KGs can result in NNs favoring the learning of common rules, which can lead to subpar performance on rare rules or exceptional cases. Furthermore, NNs may exhibit limited generalization capabilities when applying learned rules to new examples. In scenarios where examples are similar but slightly different, the model may be unable to accurately infer the correct rules.

7. Comparison and Analysis

The evolution of logical rule-based KGR reasoning methods—from ILP and the unification of probabilistic graph models with logical rules, to the combination of embeddings with logical rules, and the joint application of NNs with logical rules—highlights the continuous advancements in this field. Collectively, these categories have exhibited progressive improvements in performance. Nevertheless, each category and its corresponding sub-methods possess distinct strengths and weaknesses concerning efficiency, interpretability, and transferability.

7.1. Reasoning Accuracy

The key evaluation metrics for KGR include mean reciprocal rank (MRR), Hit@k ($k = 1, 3, 10$) denoting the proportion of accurate predictions among the top-k outcomes, and the area under the precision–recall curve (AUC-PR). Drawing upon the available experimental findings in the references, we consolidate the performance of these methods on the FB15K-237 KGR dataset [117] for link prediction tasks, as presented in Table 7.

In Table 7, if a method did not conduct experiments on FB15K-237, failed to adopt the aforementioned metrics, did not conduct dataset partitioning, or solely concentrated on a specific task (e.g., entity prediction), the corresponding values in the table are left empty. For methods that solely executed experiments on the FB15K dataset (FB15K-237 without eliminating reverse relations), a horizontal line is placed beneath the corresponding metric. In the case of methods employing the few-shot subset of the dataset, a wavy line is designated below the corresponding metric. Early methods that neither conducted experiments on benchmark datasets nor could be effectively reproduced in subsequent studies are unable to furnish experimental data. The findings from Table 7 illustrate that methods based on jointly using NNs and logical rules generally exhibit superior accuracy in comparison to the other methods. Considering the diversity of models and the inconsistency of parameter tuning methods, we do not introduce specific parameter configurations or analyses to avoid unnecessary confusion.

Table 7. Comparison of the accuracies of logical rule-based KGR methods on the FB15k-237 dataset for link prediction.

Categories	Methods	MRR	Hit@1	Hit@3	Hit@10	AUC-PR
ILP-based	AMIE+	0.292	0.174		0.409	
	RuleN		0.182		0.42	
	C-NN	0.296	0.222		0.446	
	AnyBURL	0.31	0.233			
	SARFRAN	0.389	0.298		0.537	
	Rein-AnyBURL	0.354	0.267		0.527	
Unifying probabilistic graph models and logical rules	MLN	0.098	0.067	0.103	0.16	
Unifying embedding techniques and logical rules	KALE	0.523	0.383	0.616	0.762	
	RUGE	0.768	0.703	0.815	0.865	
	TARE	0.781	0.721	0.839	0.902	
	C-NNE-AER	0.803	0.761	0.831	0.874	
Jointly using NNs and logical rules	LAN	0.394	0.302	0.446	0.566	
	VN	0.463	0.345	0.526	0.701	
	pLogicNet	0.332	0.237	0.367	0.524	
	PGAT	0.457	0.377	0.494	0.609	
	RuleGuider	0.385	0.307		0.569	
	Neural LP	0.24				
	Num-Neural LP	0.259				
	NLIL	0.25				
	DRUM	0.343	0.255		0.516	
	GCR	0.492	0.49			
	GraIL				0.641	0.847
	CoMPILE				0.676	0.855
	TACT				0.741	0.83
	ConGLR				0.683	0.857
	SNRI				0.718	0.867
	BERTRL				0.62	0.695
	RPC-IR				0.676	0.872
	CBGNN				0.975	0.963
	RNNLogic	0.349	0.258	0.385	0.533	
	CSR	0.624	0.479		0.786	

7.2. Efficiency

In approaches based on ILP, the computational complexity of FOIL-related methods grows exponentially as the number of rules increases. Consequently, their efficiency diminishes when applied to large-scale KGs. Association rule mining-based methods face challenges in dealing with large KGs due to the requirement of rule traversal. In contrast, relation path sampling-based methods demonstrate notable efficiency improvements and are suitable for rule learning in large-scale KGs.

Methods that unify probabilistic graph models and logical rules typically suffer from low efficiency due to intricate search and probability calculations, rendering them less applicable to large-scale KGs. Conversely, methods that unify embedding and logical rules mitigate the time-consuming search, traversal, and sampling procedures through representation learning techniques, leading to higher efficiency compared to the aforementioned categories. Methods that jointly use NNs and logical rules exhibit significantly higher efficiency compared to the preceding three categories of methods thanks to the efficiency and expandability of the NN model.

7.3. Interpretability

In terms of interpretability, ILP-based methods exhibit the highest level of interpretability. Conversely, methods that unify probabilistic graphical models and logical rules demonstrate comparatively lower interpretability due to their reliance on probability estimation and fuzzy reasoning. Methods that unify embeddings and logical rules, as well as methods that jointly use NNs and rules, introduces “black-box” models that are inherently implicit, thereby affecting interpretability. Within these two categories, methods that depend on rule learning for reasoning show superior interpretability.

7.4. Transferability

Transferability refers to the capacity of a model or method to leverage prior knowledge for acquiring new knowledge. In the realm of KGR methods, transferability includes the additional cost and performance variations when applying a method to diverse KGs. In general, methods that jointly use NNs and rules demonstrate the highest degree of transferability, owing to the inherent flexibility and adaptability advantages of NNs. Following closely are methods that unify embeddings and rules. Conversely, ILP-based methods and methods that unify probabilistic graphical models and logical rules exhibit relatively weaker transferability due to the necessity of relearning numerous rules when confronted with different KGs.

8. Challenges and Future Prospects

By integrating embeddings, neural networks (NNs), and other techniques, knowledge graph reasoning (KGR) approaches based on logical rules have achieved remarkable advancements. Consequently, they have exhibited encouraging performance on large-scale KGR tasks. However, despite the relatively high interpretability, reliability, and generalization capability of logical rule-based methods, there remains a gap in terms of accuracy and efficiency compared to some state-of-the-art reasoning methods solely relying on NNs. To address this, future research should focus on addressing several new challenges encountered by these reasoning methods.

Efficiency of rule learning. Despite notable advancements achieved by researchers through the integration of deep learning techniques, the predominant approach for rule learning still heavily relies on path searches within KGs. The exponential increase in the number of paths with the expansion of entities and relations exacerbates this challenge. Consequently, it is imperative to explore effective sampling strategies that can reduce the search space or establish pre-trained models capable of incorporating prior information, textual data, and corpora to mitigate the complexity of the search process. Furthermore, there is a need for further investigation into representing rules in a format that is more readily compatible with NNs, thus amplifying the effectiveness of rule learning and reasoning.

Rule quality evaluation. Rule quality evaluation traditionally relies on metrics, such as confidence and support, assuming a closed-world environment. However, these metrics pose challenges when dealing with incomplete data. To address this issue, PCA [45] relaxes the requirements for negative examples in the confidence calculation, mitigating the impact of data incompleteness. Nevertheless, PCA serves as a compromise solution rather than a definitive approach. Therefore, future research should focus on the development of more rational and practical rule evaluation metrics. Furthermore, it is crucial to investigate the systematic and in-depth impact of rule quality on the accuracy of KGR. As the primary objective of rule learning is to facilitate reasoning, understanding how rule quality influences this process is of utmost importance. Notably, there are cases where incorrect rules can still yield correct answers, warranting further exploration in this area.

Interpretability. The inherent explainability of logical rule-based KGR methods constitutes their primary advantage, contributing to heightened security and reliability in practical applications. However, these methods are impeded by their limited efficiency. The integration of NNs and embedding models with logical rules has emerged as a viable strategy for enhancing reasoning speed. Nonetheless, the attainment of model interpretability and the delicate balance between accuracy and interpretability continue to present challenges. Future advancements lie in embedding reasoning steps within the model architecture, offering an avenue to explore intermediate representations that facilitate a deeper comprehension of rules. Moreover, the effective integration of rules and embeddings can be achieved through the application of explainable machine learning techniques.

Benchmark datasets. Currently, commonly employed datasets in the KGR task, such as FB15K-237, NELL-995 [118], and WN18R [119], are employed to evaluate and compare the performance of supervised KGR methods. Nevertheless, these datasets are unable to accurately capture and represent all underlying rules and logical patterns. In some cases, erroneous or redundant rules may yield correct results, which can misguide the interpretability of the results. The application of such misleading results in critical domains can lead to severe consequences. Consequently, it is imperative to propose benchmark datasets in the future that include diverse pre-defined rules or logical patterns, enabling the simultaneous evaluation of rule learning and KGR performance.

TKG and MMKG reasoning. Currently, the majority of research studies on rule-based reasoning methods primarily focus on SKGs, with only a limited number of studies, exemplified by TLogic [120], exploring the utilization of rules for reasoning in TKGs. The process of TKG reasoning involves uncovering time-dependent or dynamic rules. Hence, it is worthwhile to explore the integration of models capable of capturing temporal information, such as RNN and LSTM, with existing rule-learning methods to acquire rules in TKGs. MMKGs include a wide range of heterogeneous information. The formulation and acquisition of rules in MMKGs exhibit a higher level of complexity and pose greater challenges. Consequently, the next step entails investigating the employment of existing multi-modal learning models to automatically extract rules from MMKGs using techniques such as data fusion, joint representation, and others.

Few-shot and zero-shot rule discovery in KGs. Rule learning poses greater challenges in few-shot, one-shot, and zero-shot learning scenarios, particularly for higher-order rules that are difficult to acquire. The difficulty in acquiring these rules significantly impacts the accuracy of reasoning. To address this issue, two main approaches can be considered in the future. The first approach focuses on enhancing the representation capability of knowledge by incorporating additional information. This augmentation can improve the quality of rules learned in few-shot scenarios. By strengthening the representation of limited samples through the integration of supplementary information, the learning of higher-quality rules becomes feasible. The second approach leverages existing high-quality KGs as priors to support rule learning and reasoning. By utilizing the abundance of high-quality samples available in these KGs, the process of rule learning and reasoning can be facilitated and enhanced.

Incorporating logical rules into LLMs. When applied in knowledge reasoning domains like intelligent question–answering and recommendation systems, are susceptible to information conflicts and confusion. This vulnerability hinders their ability to handle long-tail scenarios reliably [121] and increases the risk of hallucinations. The fundamental issue stems from the unconscious utilization of knowledge in tasks without proper guidance. Additionally, LLMs face the challenge of “knowledge scarcity” due to sparse data in domain-specific question–answering and reasoning. Therefore, an area with promising potential for future research lies in exploring the integration of KGs and rules into LLMs. This integration can enhance pre-training, during-training, and post-training stages of pre-trained language models, offering a more robust solution.

9. Conclusions

Our study offers a comprehensive overview of knowledge graph reasoning (KGR) based on logical rules. We extensively investigate related works and categorize the associated methods, while also presenting the key technologies employed in these approaches. Moreover, we conduct a comparative analysis of their strengths and weaknesses. Additionally, we analyze the primary challenges encountered by logical rule-based KGR and provide insight into future research directions.

Author Contributions: Conceptualization, Z.Z. and Q.C.; methodology, Z.Z. and Y.S.; formal analysis, Z.Z.; investigation, Z.Z. and Y.S.; resources, Z.Z.; data curation, Z.Z.; writing—original draft preparation, Z.Z.; writing—review and editing, Z.Z. and Y.S.; visualization, Z.Z.; funding acquisition, Q.C. All authors have read and agreed to the published version of the manuscript

Funding: This research was funded by the National Natural Science Foundation of China (project grant number 62073333).

Conflicts of Interest: The authors declare no conflict of interest.

References

- Li, Z.; Zhao, Y.; Zhang, Y. A Review of Knowledge Graph Reasoning Based on Representation Learning. *Chin. J. Comput. Sci.* **2023**, *50*, 94–113.
- Miller, G.A. WordNet: A Lexical Database for English. *Commun. ACM* **1995**, *38*, 39–41. [[CrossRef](#)]
- Lehmann, J.; Isele, R.; Jakob, M.; Jentzsch, A.; Kontokostas, D.; Mendes, P.N.; Hellmann, S.; Morsey, M.; van Kleef, P.; Auer, S.; et al. DBpedia—A large-scale, multilingual knowledge base extracted from Wikipedia. *Semant. Web* **2015**, *6*, 167–195. [[CrossRef](#)]
- Suchanek, F.M.; Kasneci, G.; Weikum, G. Yago: A Core of Semantic Knowledge. In Proceedings of the 16th International Conference on World Wide Web, New York, NY, USA, 18 May 2007; WWW '07; pp. 697–706. [[CrossRef](#)]
- Mitchell, T.; Cohen, W.; Hruschka, E.; Talukdar, P.; Yang, B.; Betteridge, J.; Carlson, A.; Dalvi, B.; Gardner, M.; Kisiel, B.; et al. Never-Ending Learning. *Commun. ACM* **2018**, *61*, 103–115. [[CrossRef](#)]
- Wu, W.; Li, H.; Wang, H.; Zhu, K.Q. Probbase: A Probabilistic Taxonomy for Text Understanding. In Proceedings of the ACM International Conference on Management of Data (SIGMOD), ACM International Conference on Management of Data (SIGMOD), Scottsdale, AZ, USA, 20 May 2012.
- Xu, B.; Xu, Y.; Liang, J.; Xie, C.; Liang, B.; Cui, W.; Xiao, Y. CN-DBpedia: A Never-Ending Chinese Knowledge Extraction System. In *Proceedings of the Advances in Artificial Intelligence: From Theory to Practice, Kuala Lumpur, Malaysia, 26 July 2017*; Benferhat, S., Tabia, K., Ali, M., Eds.; Springer: Cham, Switzerland, 2017; pp. 428–438.
- Niu, X.; Sun, X.; Wang, H.; Rong, S.; Qi, G.; Yu, Y. Zhishi.me-Weaving Chinese Linking Open Data. In Proceedings of the Semantic Web—ISWC 2011, Berlin/Heidelberg, Germany, 23 October 2011; pp. 205–220.
- Wang, X.; He, X.; Cao, Y.; Liu, M.; Chua, T.S. KGAT: Knowledge Graph Attention Network for Recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, New York, NY, USA, 4 August 2019; KDD '19; pp. 950–958. [[CrossRef](#)]
- Hu, S.; Zou, L.; Yu, J.X.; Wang, H.; Zhao, D. Answering Natural Language Questions by Subgraph Matching over Knowledge Graphs. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 824–837. [[CrossRef](#)]
- Zhou, H.; Young, T.; Huang, M.; Zhao, H.; Xu, J.; Zhu, X. Commonsense Knowledge Aware Conversation Generation with Graph Attention. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13 July 2018; AAAI Press: Palo Alto, CA, USA, 2018; IJCAI'18; pp. 4623–4629.
- Liang, K.; Meng, L.; Liu, M.; Liu, Y.; Tu, W.; Wang, S.; Zhou, S.; Liu, X.; Sun, F. Reasoning over Different Types of Knowledge Graphs: Static, Temporal and Multi-Modal. *arXiv* **2022**, arXiv:2212.05767. [[CrossRef](#)].

13. Guan, S.; Jin, X.; Jia, Y.; Wang, Y.; Cheng, X. Research Progress on Knowledge Reasoning for Knowledge Graphs. *Chin. J. Softw.* **2018**, *29*, 2966–2994.
14. Mavromatis, C.; Karypis, G. ReaRev: Adaptive Reasoning for Question Answering over Knowledge Graphs. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP, Abu Dhabi, United Arab Emirates, 7 December 2022; pp. 2447–2458. [\[CrossRef\]](#)
15. Xian, Y.; Fu, Z.; Muthukrishnan, S.; de Melo, G.; Zhang, Y. Reinforcement Knowledge Graph Reasoning for Explainable Recommendation. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 21 July 2019; SIGIR'19; pp. 285–294. [\[CrossRef\]](#)
16. Tuan, Y.L.; Beygi, S.; Fazel-Zarandi, M.; Gao, Q.; Cervone, A.; Wang, W.Y. Towards Large-Scale Interpretable Knowledge Graph Reasoning for Dialogue Systems. In Proceedings of the Findings of the Association for Computational Linguistics: ACL, Dublin, Ireland, 22 May 2022; pp. 383–395. [\[CrossRef\]](#)
17. Jaradeh, M.Y.; Singh, K.; Stocker, M.; Both, A.; Auer, S. Information Extraction Pipelines for Knowledge Graphs. *Knowl. Inf. Syst.* **2023**, *65*, 1989–2016. [\[CrossRef\]](#)
18. Marino, K.; Salakhutdinov, R.; Gupta, A. The More You Know: Using Knowledge Graphs for Image Classification. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 20–28. [\[CrossRef\]](#)
19. Ma, Y.; Xiang, N.; Dou, Y. Application and Research of Knowledge Graph in Military System Engineering. *Syst. Eng. Electron.* **2022**, *44*, 146–153.
20. Yuan, J.; Liu, G.; Liang, H.; Luo, Q. Summary of Research and Application of Knowledge Graphs in Risk Management Field of Commercial Banks. *Comput. Eng. Appl.* **2022**, *58*, 37–52.
21. Huang, K.; Jiang, C. Analysing and Reasoning Knowledge of Urban Transportation, Based on Ontology. *Chin. J. Comput. Sci.* **2007**, *34*, 5.
22. Hu, H.; Hao, Y.; Hao, Y.; Zhang, H. A Review of Network Security Measurement Based on Attack Graphs. *J. Cybersecur. Inf. Secur.* **2018**, *4*, 1. [\[CrossRef\]](#)
23. Nickel, M.; Murphy, K.; Tresp, V.; Gabrilovich, E. A Review of Relational Machine Learning for Knowledge Graphs. *Proc. IEEE* **2016**, *104*, 11–33. [\[CrossRef\]](#)
24. Wang, Q.; Mao, Z.; Wang, B.; Guo, L. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 2724–2743. [\[CrossRef\]](#)
25. Xia, Y.; Lan, M.; Xia, X.; Luo, J.; Zhou, G.; He, P. A Review of Explainable Knowledge Graph Reasoning Methods. *Chin. J. Cybersecur. Inf. Secur.* **2022**, *8*, 1–25. [\[CrossRef\]](#)
26. Hou, Z.; Jin, X.; Chen, J.; Guan, S.; Wang, Y.; Cheng, X. Survey of Interpretable Reasoning on Knowledge Graphs. *Chin. J. Softw.* **2022**, *33*, 4644–4667.
27. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.V.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November 2022; Volume 35, pp. 24824–24837.
28. Bian, N.; Han, X.; Sun, L.; Lin, H.; Lu, Y.; He, B. ChatGPT is a Knowledgeable but Inexperienced Solver: An Investigation of Commonsense Problem in Large Language Models. *arXiv* **2023**, arXiv:2303.16421.
29. Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; et al. PaLM: Scaling Language Modeling with Pathways. *arXiv* **2022**, arXiv:2204.02311.
30. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. A Survey of Large Language Models. *arXiv* **2023**, arXiv:2303.18223.
31. Ma, R.; Li, Z.; Chen, Z.; Zhao, L. Review of Reasoning on Knowledge Graph. *Chin. J. Comput. Sci.* **2022**, *49*, 74–85.
32. Zhang, Z.; Cao, L.; Chen, X.; Kou, D.; Song, T. Survey of Knowl Reasoning Based on Neural Network. *Comput. Eng. Appl.* **2019**, *55*, 8–19+36.
33. Sun, S.; Li, X.; Li, W.; Lei, D.; Li, S.; Yang, L.; Wu, Y. Review of Graph Neural Network Applied to Knowledge Graph Reasoning. *Chin. J. Front. Comput. Sci. Technol.* **2023**, *17*, 27–52.
34. Song, H.; Zhao, G.; Sun, R. Development of Knowledge Reasoning Based on Deep Reinforcement Learning. *Comput. Eng. Appl.* **2022**, *58*, 12–25.
35. Hitzler, P.; Janowicz, K.; Li, W.; Qi, G.; Ji, Q. Hybrid Reasoning in Knowledge Graphs: Combining Symbolic Reasoning and Statistical Reasoning. *Semant. Web* **2020**, *11*, 53–62. [\[CrossRef\]](#)
36. Zhang, W.; Chen, J.; Li, J.; Xu, Z.; Pan, J.Z.; Chen, H. Knowledge Graph Reasoning with Logics and Embeddings: Survey and Perspective. *arXiv* **2022**, arXiv:2202.07412.
37. Zhang, J.; Chen, B.; Zhang, L.; Ke, X.; Ding, H. Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open* **2021**, *2*, 14–35. [\[CrossRef\]](#)
38. Tian, L.; Zhou, X.; Wu, Y.P.; Zhou, W.T.; Zhang, J.H.; Zhang, T.S. Knowledge graph and knowledge reasoning: A systematic review. *J. Electron. Sci. Technol.* **2022**, *20*, 100159. [\[CrossRef\]](#)
39. Hao, J.; Hui, X.; Ma, S.; Jin, M. Study on Axiomatic Truth Degree in First-Order Logic. *Chin. J. Comput. Sci.* **2021**, *48*, 669–671+712.
40. Salam, A.; Schwitter, R.; Orgun, M.A. Probabilistic Rule Learning Systems: A Survey. *ACM Comput. Surv.* **2021**, *54*, 1–16. [\[CrossRef\]](#)

41. Schoenmackers, S.; Etzioni, O.; Weld, D.S.; Davis, J. Learning First-Order Horn Clauses from Web Text. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Cambridge, MA, USA, 9 October 2010; EMNLP '10; pp. 1088–1098.
42. Landwehr, N.; Kersting, K.; Raedt, L.D. Integrating Naïve Bayes and FOIL. *J. Mach. Learn. Res.* **2007**, *8*, 481–507.
43. Landwehr, N.; Passerini, A.; De Raedt, L.; Frasconi, P. Fast learning of relational kernels. *Mach. Learn.* **2010**, *78*, 305–342. [\[CrossRef\]](#)
44. Zeng, Q.; Patel, J.M.; Page, D. QuickFOIL: Scalable Inductive Logic Programming. *Proc. VLDB Endow.* **2014**, *8*, 197–208. [\[CrossRef\]](#)
45. Galárraga, L.A.; Teflioudi, C.; Hose, K.; Suchanek, F. AMIE: Association Rule Mining under Incomplete Evidence in Ontological Knowledge Bases. In Proceedings of the 22nd International Conference on World Wide Web, New York, NY, USA, 13 May 2013; WWW '13; pp. 413–422. [\[CrossRef\]](#)
46. Galárraga, L.; Teflioudi, C.; Hose, K.; Suchanek, F.M. Fast Rule Mining in Ontological Knowledge Bases with AMIE+. *VLDB J.* **2015**, *24*, 707–730. [\[CrossRef\]](#)
47. Wang, Z.; Li, J. RDF2Rules: Learning Rules from RDF Knowledge Bases by Mining Frequent Predicate Cycles. *arXiv* **2015**, arXiv:1512.07734.
48. Lajus, J.; Galárraga, L.; Suchanek, F. Fast and Exact Rule Mining with AMIE 3. In *Proceedings of the Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, 31 May–4 June 2020*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 36–52. [\[CrossRef\]](#)
49. Meilicke, C.; Fink, M.; Wang, Y.; Ruffinelli, D.; Gemulla, R.; Stuckenschmidt, H. Fine-Grained Evaluation of Rule- and Embedding-based Systems for Knowledge Graph Completion. In *Proceedings of the The Semantic Web—ISWC 2018: 17th International Semantic Web Conference, Monterey, CA, USA, 8–12 October 2018*; Proceedings, Part I; Springer: Berlin/Heidelberg, Germany, 2018; pp. 3–20. [\[CrossRef\]](#)
50. Meilicke, C.; Chekol, M.W.; Ruffinelli, D.; Stuckenschmidt, H. Anytime Bottom-up Rule Learning for Knowledge Graph Completion. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, Macao, China, 10 August 2019*; AAAI Press: Palo Alto, CA, USA, 2019; IJCAI'19; pp. 3137–3143.
51. Ferré, S. Link Prediction in Knowledge Graphs with Concepts of Nearest Neighbours. In *Proceedings of the Semantic Web, Auckland, New Zealand, 26 October 2019*; Hitzler, P., Fernández, M., Janowicz, K., Zaveri, A., Gray, A.J., Lopez, V., Haller, A., Hammar, K., Eds.; Springer: Cham, Switzerland, 2019; pp. 84–100.
52. Meilicke, C.; Chekol, M.W.; Fink, M.; Stuckenschmidt, H. Reinforced Anytime Bottom Up Rule Learning for Knowledge Graph Completion. *arXiv* **2020**, arXiv:2004.04412.
53. Ott, S.; Meilicke, C.; Samwald, M. SAFRAN: An interpretable, rule-based link prediction method outperforming embedding models. *arXiv* **2021**, arXiv:2109.08002.
54. Fetzer, J.H., Statistical Explanation and Statistical Relevance. In *Scientific Knowledge: Causation, Explanation, and Corroboration*; Springer: Dordrecht, The Netherlands, 1981; pp. 77–103. [\[CrossRef\]](#)
55. Kok, S.; Domingos, P. Learning the Structure of Markov Logic Networks. In Proceedings of the 22nd International Conference on Machine Learning, New York, NY, USA, 7 August 2005; ICML '05; pp. 441–448. [\[CrossRef\]](#)
56. De Raedt, L.; Kimmig, A.; Toivonen, H. ProbLog: A probabilistic prolog and its application in link discovery. In Proceedings of the IJCAI-07, Twentieth International Joint Conference on Artificial Intelligence, Hyderabad, India, 6 January 2007; AAAI Press International Joint Conferences on Artificial Intelligence Cop.; International Joint Conference on Artificial Intelligence; Conference date: 01-01-1800; pp. 2468–2473.
57. Cussens, J. Parameter Estimation in Stochastic Logic Programs. *Mach. Learn.* **2001**, *44*, 245–271. [\[CrossRef\]](#)
58. Wang, W.Y.; Mazaitis, K.; Cohen, W.W. Programming with Personalized Pagerank: A Locally Groundable First-Order Probabilistic Logic. In Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, New York, NY, USA, 27 October 2013; CIKM '13; pp. 2129–2138. [\[CrossRef\]](#)
59. Kimmig, A.; Bach, S.H.; Broecheler, M.; Huang, B.; Getoor, L. A short introduction to probabilistic soft logic. In Proceedings of the Neural Information Processing Systems, Stateline, NV, USA, 3 December 2012.
60. Pujara, J.; Miao, H.; Getoor, L.; Cohen, W.W. Ontology-Aware Partitioning for Knowledge Graph Identification. In Proceedings of the 2013 Workshop on Automated Knowledge Base Construction, New York, NY, USA, 27 October 2013; AKBC '13; pp. 19–24. [\[CrossRef\]](#)
61. Bach, S.H.; Broecheler, M.; Huang, B.; Getoor, L. Hinge-Loss Markov Random Fields and Probabilistic Soft Logic. *J. Mach. Learn. Res.* **2017**, *18*, 3846–3912.
62. Gallaire, H.; Minker, J.; Nicolas, J.M. Logic and Databases: A Deductive Approach. In *Readings in Artificial Intelligence and Databases*; Mylopoulos, J., Brodie, M., Eds.; Morgan Kaufmann: San Francisco, CA, USA, 1989; pp. 231–247. [\[CrossRef\]](#)
63. Cali, A.; Gottlob, G.; Lukasiewicz, T. A General Datalog-based Framework for Tractable Query Answering over Ontologies. In Proceedings of the Twenty-Eighth ACM Sigmod-Sigact-Sigart Symposium on Principles of Database Systems, New York, NY, USA, 29 June 2009; PODS '09; pp. 77–86. [\[CrossRef\]](#)
64. Lanzinger, M.; Sferrazza, S.; Gottlob, G. MV-Datalog+-: Effective Rule-based Reasoning with Uncertain Observations. *arXiv* **2022**, arXiv:2202.01718.
65. Gavalec, M.; Němcová, Z.; Sergeev, S. Tropical linear algebra with the Łukasiewicz T-norm. *Fuzzy Sets Syst.* **2015**, *276*, 131–148. [\[CrossRef\]](#)

66. Guo, S.; Wang, Q.; Wang, L.; Wang, B.; Guo, L. Jointly Embedding Knowledge Graphs and Logical Rules. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 17 October 2016; pp. 192–202. [\[CrossRef\]](#)
67. Wang, W.Y.; Cohen, W.W. Learning First-Order Logic Embeddings via Matrix Factorization. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, New York, NY, USA, 9 July 2016*; AAAI Press: Palo Alto, CA, USA, 2016; IJCAI'16; pp. 2132–2138.
68. Wang, M.; Rong, E.; Zhuo, H.; Zhu, H. Embedding Knowledge Graphs Based on Transitivity and Asymmetry of Rules. In *Proceedings of the Advances in Knowledge Discovery and Data Mining, Melbourne, VIC, Australia, 3 June 2018*; Phung, D., Tseng, V.S., Webb, G.I., Ho, B., Ganji, M., Rashidi, L., Eds.; Springer: Cham, Switzerland, 2018; pp. 141–153.
69. Guo, S.; Wang, Q.; Wang, L.; Wang, B.; Guo, L. Knowledge Graph Embedding with Iterative Guidance from Soft Rules. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, Hilton New Orleans Riverside, New Orleans, LA, USA, 2 February 2018*; AAAI Press: Palo Alto, CA, USA, 2018; AAAI'18/IAAI'18/EAAI'18.
70. Ding, B.; Wang, Q.; Wang, B.; Guo, L. Improving Knowledge Graph Embedding Using Simple Constraints. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 15 July 2018; pp. 110–121. [\[CrossRef\]](#)
71. Wang, P.; Dou, D.; Wu, F.; de Silva, N.; Jin, L. Logic Rules Powered Knowledge Graph Embedding. *arXiv* **2019**, arXiv:1903.03772.
72. Niu, G.; Zhang, Y.; Li, B.; Cui, P.; Liu, S.; Li, J.; Zhang, X. Rule-Guided Compositional Representation Learning on Knowledge Graphs. In Proceedings of the AAAI Conference on Artificial Intelligence, Hilton New York Midtown, New York, NY, USA, 7 February 2020; Volume 34, pp. 2950–2958.
73. Tang, X.; Zhu, S.C.; Liang, Y.; Zhang, M. RuleE: Neural-Symbolic Knowledge Graph Reasoning with Rule Embedding. *arXiv* **2023**, arXiv:2210.14905.
74. Ho, V.T.; Stepanova, D.; Gad-Elrab, M.H.; Kharlamov, E.; Weikum, G. Rule Learning from Knowledge Graphs Guided by Embedding Models. In *Proceedings of the The Semantic Web—ISWC, Monterey, CA, USA, 8 October 2018*; Vrandečić, D., Bontcheva, K., Suárez-Figueroa, M.C., Presutti, V., Celino, I., Sabou, M., Kaffee, L.A., Simperl, E., Eds.; Springer: Cham, Switzerland, 2018; pp. 72–90.
75. Zhang, W.; Paudel, B.; Wang, L.; Chen, J.; Zhu, H.; Zhang, W.; Bernstein, A.; Chen, H. Iteratively Learning Embeddings and Rules for Knowledge Graph Reasoning. In Proceedings of the World Wide Web Conference, New York, NY, USA, 13 May 2019; WWW '19; pp. 2366–2377. [\[CrossRef\]](#)
76. Omran, P.G.; Wang, K.; Wang, Z. An Embedding-based Approach to Rule Learning in Knowledge Graphs. *IEEE Trans. Knowl. Data Eng.* **2021**, *33*, 1348–1359. [\[CrossRef\]](#)
77. Cheng, K.; Yang, Z.; Zhang, M.; Sun, Y. UniKER: A Unified Framework for Combining Embedding and Definite Horn Rule Reasoning for Knowledge Graph Inference. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online and Punta Cana, Dominican Republic, 26 August 2021; pp. 9753–9771. [\[CrossRef\]](#)
78. Bordes, A.; Usunier, N.; Garcia-Durán, A.; Weston, J.; Yakhnenko, O. Translating Embeddings for Modeling Multi-Relational Data. In Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2, Red Hook, NY, USA, 5 December 2013; NIPS'13; pp. 2787–2795.
79. Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, E.; Bouchard, G. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33th International Conference on Machine Learning, New York City, NY, USA, 19 June 2016*; JMLR.org: BROOKLINE, MA, USA, 2016; ICML'16; pp. 2071–2080.
80. Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 26 October 2014; pp. 1724–1734. [\[CrossRef\]](#)
81. Wang, P.; Han, J.; Li, C.; Pan, R. Logic Attention Based Neighborhood Aggregation for Inductive Knowledge Graph Embedding. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, HI, USA, 27 January 2019*; AAAI Press: Palo Alto, CA, USA, 2019; AAAI'19/IAAI'19/EAAI'19; pp. 7152–7159. [\[CrossRef\]](#)
82. Sun, H.; Dhingra, B.; Zaheer, M.; Mazaitis, K.; Salakhutdinov, R.; Cohen, W. Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 2 November 2018; pp. 4231–4242. [\[CrossRef\]](#)
83. Lei, D.; Jiang, G.; Gu, X.; Sun, K.; Mao, Y.; Ren, X. Learning Collaborative Agents with Rule Guidance for Knowledge Graph Reasoning. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16 November 2020; pp. 8541–8547. [\[CrossRef\]](#)
84. Xia, Y.; Lan, M.; Luo, J.; Chen, X.; Zhou, G. Iterative rule-guided reasoning over sparse knowledge graphs with deep reinforcement learning. *Inf. Process. Manag.* **2022**, *59*, 103040. [\[CrossRef\]](#)
85. Qu, M.; Tang, J. Probabilistic Logic Neural Networks for Reasoning. In *33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019*; Curran Associates Inc.: Red Hook, NY, USA, 2019; pp. 7712–7722.

86. He, Y.; Wang, Z.; Zhang, P.; Tu, Z.; Ren, Z. VN Network: Embedding Newly Emerging Entities with Virtual Neighbors. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, New York, NY, USA, 19 October 2020; CIKM '20; pp. 505–514. [\[CrossRef\]](#)
87. Harsha Vardhan, L.V.; Jia, G.; Kok, S. Probabilistic Logic Graph Attention Networks for Reasoning. In Proceedings of the Companion Proceedings of the Web Conference 2020, New York, NY, USA, 13 May 2020; WWW '20; pp. 669–673. [\[CrossRef\]](#)
88. Zhang, Y.; Chen, X.; Yang, Y.; Ramamurthy, A.; Li, B.; Qi, Y.; Song, L. Efficient Probabilistic Logic Reasoning with Graph Neural Networks. In Proceedings of the 8th International Conference on Learning Representations, Virtual, 26 April 2020; ICLR20.
89. Rocktäschel, T.; Riedel, S. end-to-end Differentiable Proving. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 4 December 2017; NIPS'17; pp. 3791–3803.
90. Minervini, P.; Bošnjak, M.; Rocktäschel, T.; Riedel, S.; Grefenstette, E. Differentiable Reasoning on Large Knowledge Bases and Natural Language. In Proceedings of the AAAI Conference on Artificial Intelligence, Hilton New York Midtown, New York, NY, USA, 7 February 2020; Volume 34, pp. 5182–5190.
91. Minervini, P.; Riedel, S.; Stenatorp, P.; Grefenstette, E.; Rocktäschel, T. Learning Reasoning Strategies in end-to-end Differentiable Proving. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; JMLR.org: BROOKLINE, MA, USA, 2020; ICML'20; pp. 6938–6949.
92. Cohen, W.W. TensorLog: A Differentiable Deductive Database. *arXiv* **2016**, arXiv:1605.06523.
93. Yang, F.; Yang, Z.; Cohen, W.W. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 4 December 2017; NIPS'17; pp. 2316–2325.
94. Sadeghian, A.; Armandpour, M.; Ding, P.; Wang, D.Z. DRUM: End-to-end Differentiable Rule Mining On Knowledge Graphs. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates Inc.: Red Hook, NY, USA, 2019; pp. 15347–15357.
95. Wang, P.; Daria, S.; Csaba, D. Differentiable learning of numerical rules in knowledge graphs. In Proceedings of the 8th International Conference on Learning Representations, Virtual, 26 April 2020; ICLR20.
96. Dong, H.; Mao, J.; Lin, T.; Wang, C.; Li, L.; Zhou, D. Neural Logic Machines. In Proceedings of the 7th International Conference on Learning Representations, New Orleans, LA, USA, 23 December 2019; ICLR19.
97. Yang, Y.; Song, L. Learn to explain efficient via neural logic inductive learning. In Proceedings of the 8th International Conference on Learning Representations, Virtual, 26 April 2020; ICLR20.
98. Teru, K.K.; Denis, E.G.; Hamilton, W.L. Inductive Relation Prediction by Subgraph Reasoning. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13 July 2020; JMLR.org: BROOKLINE, MA, USA, 2020; ICML'20; pp. 9448–9457.
99. Chen, H.; Li, Y.; Shi, S.; Liu, S.; Zhu, H.; Zhang, Y. Graph Collaborative Reasoning. In Proceedings of the 15th ACM International Conference on Web Search and Data Mining, New York, NY, USA, 21 February 2022; WSDM '22; pp. 75–84. [\[CrossRef\]](#)
100. Mai, S.; Zheng, S.; Yang, Y.; Hu, H. Communicative Message Passing for Inductive Relation Reasoning. In Proceedings of the 35th AAAI Conference on Artificial Intelligence, Virtual, 2 February 2021; pp. 4294–4302.
101. Chen, J.; He, H.; Wu, F.; Wang, J. Topology-Aware Correlations Between Relations for Inductive Link Prediction in Knowledge Graphs. In Proceedings of the 35th AAAI Conference on Artificial Intelligence, Virtual, 2 February 2021; pp. 6271–6278.
102. Pan, Y.; Liu, J.; Zhang, L.; Hu, X.; Zhao, T.; Lin, Q. Learning First-Order Rules with Relational Path Contrast for Inductive Relation Reasoning. *arXiv* **2021**, arXiv:2110.08810.
103. Lin, Q.; Liu, J.; Xu, F.; Pan, Y.; Zhu, Y.; Zhang, L.; Zhao, T. Incorporating Context Graph with Logical Reasoning for Inductive Relation Prediction. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11 July 2022; SIGIR '22; pp. 893–903. [\[CrossRef\]](#)
104. Xu, X.; Zhang, P.; He, Y.; Chao, C.; Yan, C. Subgraph Neighboring Relations Infomax for Inductive Link Prediction on Knowledge Graphs. In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, Messe Wien, Vienna, Austria, 23–29 July 2022; Raedt, L.D., Ed.; International Joint Conferences on Artificial Intelligence Organization: Cape Town, South Africa, 2022; pp. 2341–2347. [\[CrossRef\]](#)
105. Zha, H.; Chen, Z.; Yan, X. Inductive Relation Prediction by BERT. In Proceedings of the the 36th AAAI Conference on Artificial Intelligence, Virtual, 2 February 2022; pp. 5923–5931.
106. Yan, Z.; Ma, T.; Gao, L.; Tang, Z.; Chen, C. Cycle Representation Learning for Inductive Relation Prediction. In Proceedings of the 39th International Conference on Machine Learning, Baltimore, MD, USA, 17–23 July 2022; PMLR 2022, Proceedings of Machine Learning Research; Volume 162, pp. 24895–24910.
107. Qu, M.; Chen, J.; Xhonneux, L.P.; Bengio, Y.; Tang, J. RNNLogic: Learning logic rules for reasoning on knowledge graphs. In Proceedings of the 9th International Conference on Learning Representations, Virtual Event, Austria, 3–7 May 2021; ICLR21.
108. Du, Z.; Zhou, C.; Yao, J.; Tu, T.; Cheng, L.; Yang, H.; Zhou, J.; Tang, J. CogKR: Cognitive Graph for multi-hop Knowledge Reasoning. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 1283–1295. [\[CrossRef\]](#)
109. Huang, Q.; Ren, H.; Leskovec, J. Few-shot Relational Reasoning via Connection Subgraph Pretraining. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022; Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2022; Volume 35, pp. 6397–6409.

110. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [\[CrossRef\]](#)
111. Wang, H.; Wei, Z.; Gan, J.; Wang, S.; Huang, Z. Personalized PageRank to a Target Node, Revisited. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery, New York, NY, USA, 6–10 July 2020; KDD '20; pp. 657–667. [\[CrossRef\]](#)
112. Zhang, C.; B  tepage, J.; Kjellstr  m, H.; Mandt, S. Advances in Variational Inference. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 2008–2026. [\[CrossRef\]](#)
113. Zhao, Y.; Qi, J.; Liu, Q.; Zhang, R. WGCN: Graph Convolutional Networks with Weighted Structural Features. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 11 July 2021; SIGIR '21; pp. 624–633. [\[CrossRef\]](#)
114. Gallaire, H.; Minker, J.; Nicolas, J.M. Logic and Databases: A Deductive Approach. *ACM Comput. Surv.* **1984**, *16*, 153–185. [\[CrossRef\]](#)
115. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 4 December 2017; NIPS'17; pp. 6000–6010.
116. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2 June 2019; pp. 4171–4186. [\[CrossRef\]](#)
117. Toutanova, K.; Chen, D. Observed versus latent features for knowledge base and text inference. In Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, Beijing, China, 15 July 2015; pp. 57–66. [\[CrossRef\]](#)
118. Demir, C.; Ngomo, A.C.N. Convolutional Complex Knowledge Graph Embeddings. In Proceedings of the Semantic Web, Virtual Event, 24–28 October 2021; Verborgh, R., Hose, K., Paulheim, H., Champin, P.A., Maleshkova, M., Corcho, O., Ristoski, P., Alam, M., Eds.; Springer: Cham, Switzerland, 2021; pp. 409–424.
119. Xiong, W.; Hoang, T.; Wang, W.Y. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7 September 2017; pp. 564–573. [\[CrossRef\]](#)
120. Liu, Y.; Ma, Y. TLogic: Temporal Logical Rules for Explainable Link Forecasting on Temporal Knowledge Graphs. In Proceedings of the 36th AAAI Conference on Artificial Intelligence, Vancouver, Canada, 2 February, 2022; pp. 4120–4127.
121. Bang, Y.; Cahyawijaya, S.; Lee, N.; Dai, W.; Su, D.; Wilie, B.; Lovenia, H.; Ji, Z.; Yu, T.; Chung, W.; et al. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. *arXiv* **2023**, arXiv:2302.04023.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.