Fundamental Research 1 (2021) 565–573

Contents lists available at ScienceDirect

Fundamental Research

journal homepage: http://www.keaipublishing.com/en/journals/fundamental-research/

Article

Neural symbolic reasoning with knowledge graphs: Knowledge extraction, relational reasoning, and inconsistency checking

Huajun Chen[a,b,c,*], Shumin Deng[a,c], Wen Zhang[a,c], Zezhong Xu[a,c], Juan Li[a,c], Evgeny Kharlamov[d,e]

[a] College of Computer Science, Zhejiang University, Hangzhou, 310027, China
[b] Hangzhou Innovation Center, Zhejiang University, Hangzhou, 310000, China
[c] AZFT Joint Lab for Knowledge Engine, Hangzhou, 311121, China
[d] Bosch Center for Artificial Intelligence, Robert Bosch GmbH, Renningen, 70465 Stuttgart, Germany
[e] SIRIUS Research Center, University of Oslo, Oslo, N-0316, Norway

ARTICLE INFO

Article history:
Received 25 April 2021
Received in revised form 12 August 2021
Accepted 20 August 2021
Available online 9 September 2021

Keywords:
Neural symbolic reasoning
Knowledge graph
Knowledge extraction
Relational reasoning
Inconsistency checking

ABSTRACT

Knowledge graphs (KGs) express relationships between entity pairs, and many real-life problems can be formulated as knowledge graph reasoning (KGR). Conventional approaches to KGR have achieved promising performance but still have some drawbacks. On the one hand, most KGR methods focus only on one phase of the KG lifecycle, such as KG completion or refinement, while ignoring reasoning over other stages, such as KG extraction. On the other hand, traditional KGR methods, broadly categorized as symbolic and neural, are unable to balance both scalability and interpretability. To resolve these two problems, we take a more comprehensive perspective of KGR with regard to the whole KG lifecycle, including KG extraction, completion, and refinement, which correspond to three subtasks: knowledge extraction, relational reasoning, and inconsistency checking. In addition, we propose the implementation of KGR using a novel neural symbolic framework, with regard to both scalability and interpretability. Experimental results demonstrate that our proposed methods outperform traditional neural symbolic models.

1. Introduction

Knowledge graphs (KGs) are collections of facts that express relationships between entity pairs or assign types to entities in the form of (*subject, predicate, object*) or (h, r, t). Many large KGs, including Freebase [1], Wikidata [2], AliMeKG [3], and OpenKG[1], have been built in recent years, and have been used for a broad range of important applications, such as question answering [4], knowledge-based language models [5], and recommender systems [6]. Reasoning with KGs, or KG reasoning (KGR), revolves around the lifecycle of KG, including *KG extraction, KG completion*, and *KG Refinement*. Therefore, KG extraction corresponds to knowledge extraction, which aims to establish structured KGs from unstructured corpora [7]. KG completion corresponds to relational reasoning, which focuses on inferring new knowledge based on existing knowledge to complete KGs [8]. KG refinement corresponds to inconsistency checking, which aims to detect noise in KGs and clean up KGs. We consider that the three stages of KGR follow the same general neural symbolic integration framework, as shown in Fig. 1: Symbolic facts and rules/axioms could be embedded into neural space, and the computation in neural space could help reasoning tasks in KGs, either by inferring new triples or by detecting noise implicitly or explicitly.

Conventional approaches to KGR can be broadly classified into two main categories: *symbolic methods* based on logical inference and *neural methods* based on vector space representations. **Symbolic methods** perform logical inference by using logical rules that are generally more precise and interpretable, thereby providing valuable insight to inference results. For example, traditional reasoners such as Pellet [9] and HermiT[2] reason over KG based on a predefined ontology, NELL [10] uses a collection of restricted Horn Clause rules to infer new belief triples, and AMIE [11] considers the problems of learning models composed of a collection of first-order logical rules. Although symbolic methods perform well in terms of precision and interpretability, they require manually defining the reasoning logic, thus lacking scalability. **Neural methods** learn latent representations of KG entities and relations in continuous vector space that are called *embeddings*. Thus, neural methods that perform embedding-based reasoning are more powerful when there are a large number of relations or triples. Neural methods such as TransE [8], HolE [12], and ComplEx [13] can preserve the information and semantics in KGs, including the existence of triples and similarities between entities. However, although neural methods are good at scalability, they

* Corresponding author.
E-mail address: huajunsir@zju.edu.cn (H. Chen).
[1] http://www.openkg.cn/
[2] http://www.hermit-reasoner.com

https://doi.org/10.1016/j.fmre.2021.08.013
2667-3258/© 2021 The Authors. Publishing Services by Elsevier B.V. on behalf of KeAi Communications Co. Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

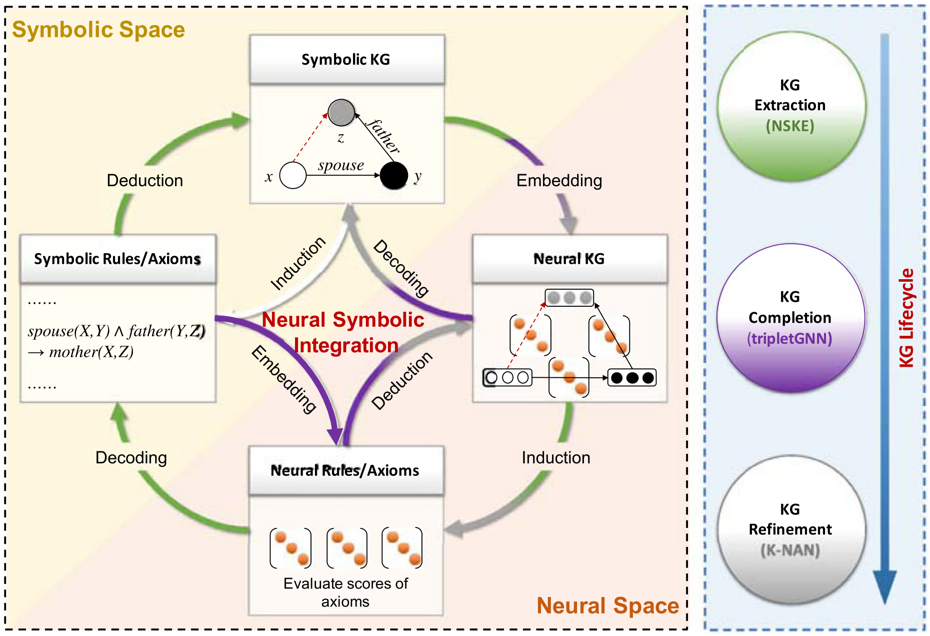

Fig. 1. Overview of the general neural symbolic integration framework with three substructures for three subtasks in KG lifecycle. Symbolic KG and Symbolic Rules/Axioms respectively store symbolic facts and rules/axioms in *Symbolic Space*, and **Neural KG** and **Neural Rules/Axioms** respectively embed these symbols in *Neural Space. KG Extraction* via NSKE (§2), *KG Completion* via tripletGNN (§3), and *KG Refinement* via K-NAN (§4), respectively, integrates some of the four modules via embedding, induction, decoding and deduction. For example, *KG Extraction* via NSKE follows the cycle of Symbolic KG $\xrightarrow{embedding}$ Neural KG $\xrightarrow{induction}$ Neural Rules/Axioms $\xrightarrow{decoding}$ Symbolic Rules/Axioms $\xrightarrow{deduction}$ Symbolic KG. Best viewed in color.

lack interpretability, and, moreover, their inference results cannot be explained.

To overcome the drawbacks of symbolic and neural methods, we propose **neural symbolic methods** for KGR, which integrate the advantages of logical inference and neural embedding, focusing on both interpretability and scalability. Previous neural symbolic models such as NeuralLP [14] learn numerical rules under the framework of differentiable rule learning, models such as IterE [15] simultaneously learn rules and embedding based on iterative learning, and models such as JOIE [16] jointly learn KG embeddings of instances and ontological concepts. However, most of them only focus on one phase in the lifecycle of KGs, such as KG completion or refinement, while ignoring other important KGR subtasks, such as KG extraction. In this study, we propose to perform **neural symbolic reasoning with KGs** in the entire lifecycle of KGs, which corresponds to three subtasks: (1) *neural symbolic reasoning for knowledge extraction*, (2) *neural symbolic relational reasoning*, and (3) *neural symbolic inconsistency checking.* Our contributions can be summarized as follows:

- We take a more comprehensive perspective of neural symbolic reasoning with KGs, and consider the entire lifecycle of KGs, including KG extraction, KG completion, and KG refinement.
- We propose a novel neural symbolic framework with three substructures for KG reasoning, which integrates the interpretability of symbolic methods and the scalability of neural methods for three subtasks: knowledge extraction (§2), relational reasoning (§3), and inconsistency checking (§4).
- We explore datasets focused on neural symbolic reasoning with KGs, and the experimental results of all subtasks demonstrate that our proposed neural symbolic framework achieves better performance than baselines.

2. Neural symbolic reasoning for knowledge extraction

In this section, we introduce neural symbolic reasoning in KG extraction, corresponding to the substructure for knowledge extraction in our proposed neural symbolic framework, as shown in Fig. 1: Symbolic KG $\xrightarrow{embedding}$ Neural KG $\xrightarrow{induction}$ Neural Rules/Axioms $\xrightarrow{decoding}$ Symbolic Rules/Axioms $\xrightarrow{deduction}$ Symbolic KG.

2.1. Task formulation

Knowledge Extraction. In this study, we consider the knowledge extraction (KE) tasks of relation extraction (RE) and event extraction (EE). Suppose that we have a relation class set $\mathcal{R} = \{r_i | i \in [1, N]\}$, an event class set $\mathcal{E} = \{e_i | i \in [1, N]\}$, and corpus $\mathcal{T} = \{X_i | i \in [1, M]\}$ that contain M instances. Each instance X_i in $\mathcal{T}$ is denoted as a token sequence $X_i = \{x_i^j | j \in [1, L]\}$ with a maximum of L tokens. Our goal is to predict the relation and event labels for each instance in the corresponding corpus. Traditional approaches to KE are mostly based on neural networks [17–24], and ignore the correlation knowledge of relation and event classes.

Neural Symbolic Reasoning for Knowledge Extraction. Neural symbolic reasoning utilizes correlation knowledge between the relation classes in $\mathcal{R}$ or the event classes in $\mathcal{E}$ for KE. We utilize semantic connections among relations for neural symbolic reasoning in RE, including implicit semantic connection with KG embedding, *e.g.*, the relation *place_lived* is more relevant to *nationality* than *profession*, and explicit semantic connection with rule learning, *e.g.*, the rule is *located_in_country(x,y) ∧ next_to_body_of_water(x,z) → basin_country_of(y,z)*. We also utilize a multi-faceted event correlation neural symbolic reasoning in EE, including event temporal and causal relations.

2.2. Model and method

In our proposed neural symbolic framework, a general substructure called *NSKE* is devised to implement neural symbolic reasoning for knowledge extraction, including three modules: (1) feature representation, (2) rule learning, and (3) correlation inference. Fig. 2 shows the key concepts of the three modules.

Feature representation aims at capturing syntax features of each instance and obtaining encoding $\boldsymbol{X}_i$ for input tokens X_i into the feature space. *Rule learning* aims to map class set $\mathcal{R}$ and $\mathcal{E}$ into a semantic space,

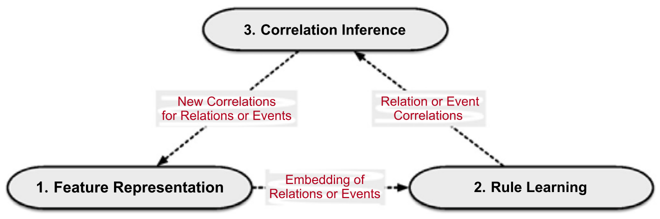
3. Correlation Inference
New Correlations for Relations or Events
Relation or Event Correlations
1. Feature Representation
Embedding of Relations or Events
2. Rule Learning

Fig. 2. Overview of our proposed NSKE.

and establish the correlation among the class set. *Correlation inference* seeks to infer new class correlations based on existing relation correlations C^r or event correlations C^e.

2.2.1. Feature representation

Given a token sequence $X_i = \{x_i^1, \cdots, x_i^L\}$, we use an instance encoder to obtain contextual representation $\boldsymbol{X}_i$. Note that the instance encoder is pluggable, for example, it can be a pre-trained BERT [25], where the token embedding of [CLS] is treated as the contextual representation $\boldsymbol{X}_i$. Furthermore, it can also be replaced by other models, such as the PCNN [26] and JRNN [27]. In addition, the entity pair for RE and the trigger for EE will be marked.

2.2.2. Rule learning

We then project $\boldsymbol{X}_i$, $\mathcal{R}$, and $\mathcal{E}$ into the semantic space. The semantic representation for $\boldsymbol{X}_i$ is regarded as the embedding of a possible corresponding class, denoted by $\boldsymbol{P}_i = \mathcal{S}(\boldsymbol{X}_i)$. The semantic projector $\mathcal{S}(\cdot)$ is pluggable, *e.g.*, it can be DeViSE [28] with a linear function or ConSE [29] with a convex combination.

The class embedding $\boldsymbol{P}_{rl}$ for $\mathcal{R}$ and $\mathcal{E}$ is calculated using a rule-guided class encoder. In KG, logic rules show connections between relations or events. They are in the form of a *body* → *head*, where *head* is a binary atom and *body* is a conjunction of binary and unary atoms, such as rule *spouse(x,y)* ∧ *father(y,z)* → *mother(x,z)*. For RE, we adopt typical rule mining methods such as AMIE [11] to generate rules from structural KGs. For EE, we adopt event-event relation recognition models such as TCR [30] to detect event correlation rules. The class embedding $\boldsymbol{P}_{rl}$ is denoted by:

$$\boldsymbol{P}_{rl} = \frac{\sum_{j=1}^{K} conf_j * \mathbb{E}(Rule_j)}{\sum_{j=1}^{K} conf_j} \tag{1}$$

where $Rule_j$ is the j_{th} rule with the top K highest confidence score, and $conf_j$ represents the confidence of $Rule_j$. $\mathbb{E}(Rule_j)$ is the embedding of $Rule_j$, which can be obtained via KG embedding models, such as DistMult [31].

2.2.3. Correlation inference

Given the correlation rules among relation or event types, we infer new correlations based on existing ones. Specifically, we utilize grounding g to infer new correlation triples, which can be generalized in the following form:

$$(h^I, r^I, t^I) \leftarrow (h^1, r^1, t^1), \cdots, (h^n, r^n, t^n) \tag{2}$$

where the right-side event triples $(h^k, r^k, t^k) \in \mathcal{G}$ with $k \in [1, n]$ already exists in KG $\mathcal{G}$ and $(h^I, r^I, t^I) \notin \mathcal{G}$ is newly inferred triples to be added.

To compute the truth value of the grounding g, we select three object properties (OP) of relations defined in OWL2[3]: subOP, inverseOP, and transitiveOP, and then learn the matrices of relations from the linear map assumption [15].

Assuming that $\boldsymbol{M}_r^{\dagger}$ and $\boldsymbol{M}_r^{\ddagger}$ denote the relation set on the left and right of Eq (2), respectively, they are matrices either from a single matrix or a product of two matrices. The normalized truth value $\mathcal{F}_p$ of g can be calculated by:

$$\mathcal{F}_p' = \|\boldsymbol{M}_r^{\dagger} - \boldsymbol{M}_r^{\ddagger}\|_F, \quad \mathcal{F}_p = \frac{\mathcal{F}_p^{max} - \mathcal{F}_p'}{\mathcal{F}_p^{max} - \mathcal{F}_p^{min}} \tag{3}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, and the subscript p denotes one of the three object properties. $\mathcal{F}_p^{\max}$ and $\mathcal{F}_p^{\min}$ are the maximum and minimum Frobenius norm scores, respectively. $\mathcal{F}_p \in [0, 1]$ is the truth value for grounding g, and a higher $\mathcal{F}_p$ indicates that g is more likely to be valid.

3. Neural symbolic relational reasoning

In this section, we introduce neural symbolic reasoning in KG completion, corresponding to the substructure for relational reasoning in our proposed neural symbolic framework. As seen in Fig. 1: $\left\{\text{Symbolic KG} \xrightarrow{embedding}; \text{Symbolic Rules/Axioms} \xrightarrow{embedding} \text{Neural Rules/Axioms} \xrightarrow{deduction}\right\}$ $\text{Neural KG} \xrightarrow{decoding} \text{Symbolic KG}$.

3.1. Problem formulation

Relational Reasoning. As one of the most important tasks in KG, its goal is to infer potential relations between entities or missing entities. Addressing this issue has become an important topic, and many approaches have emerged for knowledge graph completion (KGC). Symbolic methods provide a transparent reasoning process over KG, but lack the generalization ability to unseen examples. Recently, several studies have viewed this as a link prediction problem and attempted to solve it with some network embedding approaches, although these approaches are difficult to explain.

Neural-symbolic Relational Reasoning. We focused on combining the two reasoning methods based on symbols and neural networks. Given a KG $\mathcal{G} = \{\mathcal{E}, \mathcal{P}, \mathcal{F}, \mathcal{A}\}$, with $\mathcal{E}, \mathcal{P}, \mathcal{F}$, and $\mathcal{A}$ as the set of entities, properties, facts, and axioms. Table 1 shows the axioms and their semantics used. In this study, we aim to devise a neural axiomatic reasoning framework that not only learns embedding from facts but also utilizes ontological axioms. More precisely, our aim is to devise a neural axiomatic reasoning framework that not only learns embedding from facts but also utilizes ontological axioms in a unified and model-agnostic manner without an explicit deductive reasoning process and ad-hoc model design for different axioms.

3.2. Model and method

There are two principal challenges in neural axiomatic reasoning, with mixed utilization of facts and axioms. The first challenge is *axiom injection*. As axioms and facts are in different forms, it is difficult to inject axiomatic logic into embeddings learned from facts that are typically represented as directed labeled entity graphs. The second challenge is *model-agnosticism* for axiom utilization. As the utilization of axioms varies, we need to design ad-hoc models for different axioms that are tedious, as there are many types of axioms.

In our proposed neural symbolic framework, a substructure called tripletGNN was devised to implement neural symbolic relational reasoning. First, we propose a *triplet graph* that represents a KG with facts and axioms as an undirected unlabeled graph in contrast with conventional directed labeled *entity graph*, and a novel way to store triplet graphs instead of using adjacent matrices to better adapt the inductive graph prediction task for KGC. Second, we propose a method based on a *graph neural network* [32] enhanced with a novel node feature interaction module, in which an interaction matrix is calculated to distinguish the reason for linkage, to tackle the second challenge.

[3] https://www.w3.org/TR/owl2-profiles/

Table 1
Six OWL2 axioms, their semantics in terms of First Order Logics, and the corresponding triplets.

Axiom	Semantics	Triplet form
$OPDomain(p\,c)$	$\forall x,y: OPDomain(p\,c)$, implies $x \in c$	$(p, domain, c)$
$OPRange(p\,c)$	$\forall x,y: OPRange(p\,c)$, implies $y \in c$	$(p, range, c)$
$SubOPof(p_1\,p_2)$	$\forall x,y: SubOPof(p_1\,p_2)$ and $p_1(x\,y)$ implies $p_2(x\,y)$	$(p_1, subPropertyOf, p_2)$
$InverseOPof(p_1\,p_2)$	$\forall x,y: InverseOPof(p_1\,p_2)$ and $p_1(x\,y)$ implies $p_2(y\,x)$	$(p_1, inversePropertyOf, p_2)$
$EquivalentOP(p_1\,p_2)$	$\forall x,y: EqualOPof(p_1\,p_2)$ and $p_2(x\,y)$ implies $p_1(x\,y)$	$(p_1, equivalentPropertyOf, p_2)$
	$\forall x,y: EqualOPof(p_1\,p_2)$ and $p_1(x\,y)$ implies $p_2(x\,y)$	
$SymmetricOP(p)$	$\forall x,y: SymmetricOP(p)$ and $p(x\,y)$ implies $p(y\,x)$	$(p, symmetric, p)$

3.2.1. Triplet graph

Construction. First, we add each fact $f \in \mathcal{F}$ and axiom $a \in \mathcal{A}$ as a node into a triplet graph with their elements (e^1, e^2, e^3) as node features. Then, we add edges based on the element-sharing principle. Specifically, for the triplet pair $\{(t_i, t_j)|\{i, j\} \in [0, n_t - 1] \wedge i \neq j\}$, if e' is an element of both t_i and t_j, we add an undirected edge between them.

The intuition is that two triplets are related if they have common elements that they both depict. In the contest of the KG, unlike the line graph constructed from the entity graph, the triplet graph makes axioms into consideration, which cannot be represented in an entity graph.

Storage. We apply an alternative method to automatically and dynamically build a local triplet graph for a given triplet. We assign a unique id for each triplet and element, including entities, properties, and axiom types, in the range of $[0, n_t - 1]$ and $[0, n_e - 1]$, where n_t and n_e are the number of triplets and elements. We store two key information for a triplet graph:

$$t2e \in \mathbb{R}^{n_t \times 3},\ e2t \in \mathbb{R}^{n_e} \times m, \tag{4}$$

where $t2e$ is the triplet element mapping matrix, and $e2t$ is the element-triplet mapping matrix, where $e2t_i \in \mathbb{R}^{1\times m}$ with each cell as the id of a triplet containing e_i. where m is a hyperparameter.

Walking. Walks on the triplet graph based on a target triplet $t = (e_i, e_j, e_k)$, where i, j, k are id of the first, second, and third elements, respectively, can be obtained as follows:

$$N(t) = walk(t) = e2t_i \diamond e2t_j \diamond e2t_k, \tag{5}$$

where $e2t_x$ is the x_{th} row of the $e2t$ matrix and $x \diamond y$ denotes the concatenation of x and y; thus, $N(t) \in \mathbb{R}^{1\times 3m}$. Thus, a triplet graph of a target fact can be built on the fly given its elements.

3.2.2. TripletGNN

With a triplet graph, we propose a general neural axiomatic reasoning framework to utilize and learn from axioms and facts. Each triplet node has *node feature embedding* and *node structure embedding*, which are used to predict a target triplet. For each element e in the triplet graph, we learn an embedding $\mathbf{e} \in \mathbb{R}^d$. For each triplet $t = (e^1, e^2, e^3)$, $\mathbf{e}^1, \mathbf{e}^2, \mathbf{e}^3$ in order of feature embedding $\mathbf{t}^f \in \mathbb{R}^{3\times d}$ and learn a node structure embedding matrix $\mathbf{t}^s \in \mathbb{R}^{3\times d}$ for t. Given a target triplet $t = (e^1, e^2, e^3)$, we first obtain its neighbor triplets $N(t)$ based on the walking function defined in Eq. (5). Then, sample k from them is denoted as $\hat{N}(t)$, where $|\hat{N}(t)| = k$. The reason we choose k neighbors rather than all of the neighbor nodes is that there might be too many neighbors for a specific in some special cases, and by doing this, we can reduce the training complexity and storage space.

Triplet Feature Interaction Module. To distinguish the connection types between the two triplets, we consider feature interaction. The overview of interaction module are shown in Fig. 3, where for each sampled neighbor triplet $n \in \hat{N}(t)$, we compute an interaction matrix based on the vector cosine similarity:

$$\mathbf{I} = \frac{\mathbf{t}^f}{\|\mathbf{t}^f\|}^{\top} \times \frac{\mathbf{n}^c}{\|\mathbf{n}^c\|},\ \mathbf{n}^c = \mathbf{n}^f \diamond \mathbf{n}^s, \tag{6}$$

where $\mathbf{t}^f \in \mathbb{R}^{3\times d}$ is node feature embedding for t. $\{\mathbf{n}^f, \mathbf{n}^s\} \in \mathbb{R}^{3\times d}$ are node feature and structure embedding for n.

To utilize the interaction matrix, we learn a transformation tensor $\mathbf{T} \in \mathbb{R}^{n_e \times 18\times 18}$ with $\mathbf{T}_x \in \mathbb{R}^{18\times 18}$ as the transformation matrix for the x_{th} element e_x. Because the shape of the interaction matrix is 3×6 (three feature embedding of the target triplet interacted with three feature embeddings and three structure embeddings), after being flattened and before transformation, the shape of the interaction matrix will be 1×18. To keep the shape of the interaction matrix the same to ensure that it could be reshaped back to 3×6 again after being transformed, the shape of the transformation tensor T has to be set as 18×18. Then we transform $\mathbf{I}$: $\mathbf{I}^t \leftarrow reshape^{3\times 6}(flatten^{1\times 18}(\mathbf{I}) \times \mathbf{T}_j)$. In the transformed interaction matrix $\mathbf{I}^t$, each cell $\mathbf{I}^t_{xy}$ indicates the importance of the y_{th} vector from the neighbor triplet n to the x_{th} element from the target triplet t. Then, we generate the structure embedding of t under n by aggregating $\mathbf{n}^c$ supervised by $\mathbf{I}^t$: $\mathbf{t}^{s_n} = \mathbf{I}^t \times \mathbf{n}^c$.

Score and Loss Function. After obtaining $\mathbf{t}^{s_n}$ for each sampled neighbor $n \in \hat{N}(t)$, the final representation for t is

$$\mathbf{t} = \mathrm{MEAN}(\mathbf{t}^f, RELU(\mathrm{MEAN}(\{\mathbf{t}^f\} \cup \{\mathbf{t}^{s_n}, \forall n \in \hat{N}(t)\}) \times \mathbf{W})). \tag{7}$$

With $\mathbf{t}$, we apply the score function and loss function from KG embedding methods to calculate the score of t, which indicates its true value. The score functions are as follows:

$$f(t)_{DistMult} = \mathbf{t}_1 \mathbf{W}_{\mathbf{t}_2} \mathbf{t}_3^{\top},\ f(t)_{ComplEx} = RE(\mathbf{t}_1 \mathbf{W}_{\mathbf{t}_2} \bar{\mathbf{t}}_3^{\top}). \tag{8}$$

where $\mathbf{W}_{\mathbf{t}_2}$ is a diagonal matrix with diagonal values from $\mathbf{t}_2$ on the diagonal. $\bar{\mathbf{t}}_3$ is the conjugate of $\mathbf{t}_3$. During training, we apply cross entropy loss function with logistic activation.

4. Neural symbolic inconsistency checking

In this section, we introduce neural symbolic reasoning in KG refinement, corresponding to the substructure for inconsistency checking in our proposed neural symbolic framework. As shown in Fig. 1: Symbolic KG $\xrightarrow{embedding}$ Neural KG $\xrightarrow{induction}$ Neural Rules/Axioms $\xrightarrow{deduction}$ Neural KG $\xrightarrow{decoding}$ Symbolic KG.

4.1. Task formulation

Inconsistency Checking. It has been proposed to detect inconsistent knowledge, and the goal is to detect incorrect triples in KGs and the specific inconsistencies in these triples. Previous studies mainly focused on statistics-based methods [33,34] that exploit statistical distributions, or logic-based methods [35–37], where ontologies are utilized.

Neural Symbolic Inconsistency Checking. In this study, we used neural axiom networks and KG embedding methods for inconsistency checking. Given a noisy KG $\mathcal{G}' = \{(h', r', t') \cup (h, r, t)|(h, r, t) \in \mathcal{G}, (h', r', t') \notin \mathcal{G}\}$, where triples in $\mathcal{G}$ are correct triples, and (h', r', t') is an incorrect triple. We design a framework that can model both the structural information of triples and axiom information implied in triples. The axioms considered include domain, range, disjoint, irreflexive, and asymmetric. Our aim is to detect erroneous triples in or would be added to $\mathcal{G}'$.

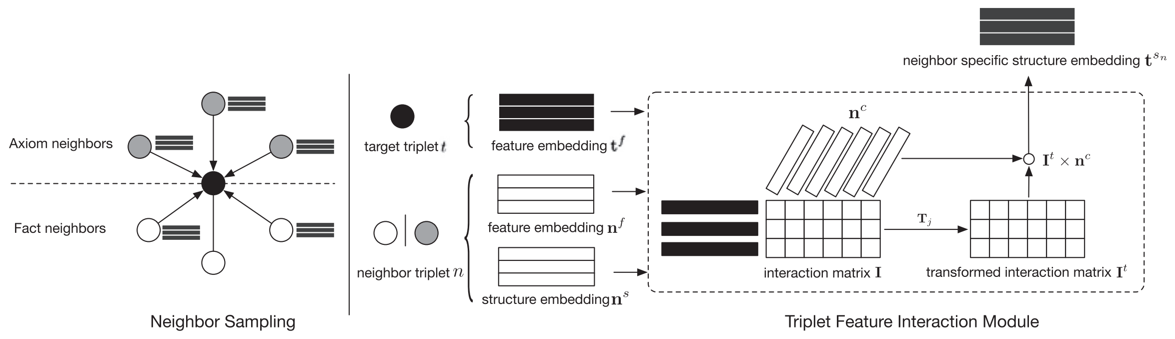
Axiom neighbors
Fact neighbors
Neighbor Sampling
target triplet t
feature embedding $\mathbf{t}^f$
neighbor triplet n
feature embedding $\mathbf{n}^f$
structure embedding $\mathbf{n}^s$
$\mathbf{n}^c$
$\mathbf{I}^t \times \mathbf{n}^c$
neighbor specific structure embedding $\mathbf{t}^{s_n}$
$\mathbf{T}_j$
interaction matrix $\mathbf{I}$
transformed interaction matrix $\mathbf{I}^t$
Triplet Feature Interaction Module

Fig. 3. Triplet feature interaction module.

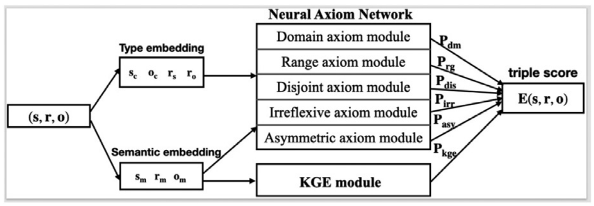

Fig. 4. Framework of our proposed substructure of K-NAN.

4.2. Model and method

In our proposed neural symbolic framework, a substructure called K-NAN was devised to implement neural symbolic inconsistency checking. K-NAN consists of a KG embedding module and a neural axiom network with five inconsistency axiom modules, as shown in Fig. 4. For a triple (h, r, t), we use $\boldsymbol{h_c}$, $\boldsymbol{t_c}$, $\boldsymbol{h_m}$, and $\boldsymbol{t_m}$ to respectively represent the type (c) and semantic (m) embeddings of h and t. $\boldsymbol{r_m}$, $\boldsymbol{r_h}$, and $\boldsymbol{r_t}$ respectively denote the semantic embedding of r, and the subject as well as object type embeddings expected by r.

4.2.1. Knowledge graph embedding

We select TransE [8] for KG embedding, as it is the simplest translation-based model, which interprets relations as translating operations between subject and object entities. The fitness of a triple is calculated as follows:

$$f_r(h, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{l_1/l_2} \tag{9}$$

where l_1 and l_2 denote the L_1 and L_2 norms, respectively. A smaller value indicates higher fitness for a triple.

4.2.2. Neural axiom network

It is intuitive that a correct triple is also a logically consistent triple, namely, the triple satisfies domain, range, disjoint, irreflexive, and asymmetric axioms.

Domain axiom consistency focuses on type compatibility between $\boldsymbol{h_c}$ and $\boldsymbol{r_h}$. To obtain a more accurate representation of $\boldsymbol{r_h}$, we consider the relations of the subject entity $\mathcal{R}(h) = \{r_i | (h, r_i, e) \in \mathcal{G}\}$, where e denotes any one entity. We introduce the attention mechanism and generate $\hat{\boldsymbol{r_h}}$ based on the relations in $\mathcal{R}(h)$. For each relation $r_i \in \mathcal{R}(h)$, the attention weight a_i is calculated, and softmax is applied over a_i:

$$a_i = f(\boldsymbol{r_h}, \boldsymbol{r_i}) = \boldsymbol{r_h}^T \boldsymbol{r_i}, r_i \in \mathcal{R}(h); \quad p_i = \frac{exp(a_i)}{\sum_{r_j \in \mathcal{R}(h)} exp(a_j)}. \tag{10}$$

where j means the j_{th} relation of h. Then, $\hat{\boldsymbol{r_h}}$ and the likelihood P_{dm} of h, r satisfying the domain axiom are defined as:

$$\hat{\boldsymbol{r_h}} = \sum_{r_i \in \mathcal{R}(h)} p_i \boldsymbol{r_i}, \ P_{dm} = f(\boldsymbol{h_c}, \boldsymbol{r_h}) = \sigma(\boldsymbol{h_c} \cdot \hat{\boldsymbol{r_h}}) \tag{11}$$

Range axiom consistency focuses on type compatibility between $\boldsymbol{r_t}$ and $\boldsymbol{t_c}$. Similarly, we devise an attention mechanism as in domain axiom consistency. We denote the relations connected to t as $\mathcal{R}(t) = \{r_i | (e, r_i, t) \in \mathcal{G}\}$ and generate $\hat{\boldsymbol{r_t}}$ based on all relations in $\mathcal{R}(t)$. The attention weight b_i and b_i after softmax applied q_i are calculated by:

$$b_i = f(\boldsymbol{r_t}, \boldsymbol{r_i}) = \boldsymbol{r_t}^T \boldsymbol{r_i}, r_i \in \mathcal{R}(t); \quad q_i = \frac{exp(b_i)}{\sum_{r_k \in \mathcal{R}(t)} exp(b_k)}. \tag{12}$$

where k denotes the k_{th} relation in the connected relations of the object entity. The generated embedding $\hat{\boldsymbol{r_t}}$, and the compatibility probability P_{rg} for satisfaction of the range axiom is defined as:

$$\hat{\boldsymbol{r_t}} = \sum_{r_i \in \mathcal{R}(t)} q_i \boldsymbol{r_i}, \ P_{rg} = f(\boldsymbol{t_c}, \boldsymbol{r_t}) = \sigma(\boldsymbol{t_c} \cdot \hat{\boldsymbol{r_t}}) \tag{13}$$

Disjoint axiom consistency focuses on the compatibility of semantic embeddings of two relations with same subject and object entities. For each triple (h, r, t), we simply copy the idea from TransE, which holds the view that $h + r = t$. Therefore, $(t_m - h_m)$ is regarded as the general representation of the relation with h and t as the subject and object entity. The disjoint axiom can be simplified to calculate the compatibility score of $(\boldsymbol{t_m} - \boldsymbol{h_m})$ and $\boldsymbol{r_m}$, which is defined as:

$$P_{dis} = f(\boldsymbol{r_m}, (\boldsymbol{t_m} - \boldsymbol{h_m})) = \sigma(\boldsymbol{r_m} \cdot (\boldsymbol{t_m} - \boldsymbol{h_m})). \tag{14}$$

Irreflexive axiom consistency focuses on whether r is irreflexive, and whether $h = t$. A triple (h, r, t) is irreflexive inconsistency if and only if r is irreflexive and $h = t$. We first determine whether $h = t$. When $h = t$, we then consider the irreflexive properties of the relation. To determine whether r is irreflexive, we apply a feed-forward layer. Besides, the type constraint $\sigma(r_t \cdot r_h)$ is also considered. The probability that a triple satisfies the irreflexive axiom is defined as:

$$P_{irr} = \begin{cases} 1, \ h \neq t \\ \sigma(W_1 \boldsymbol{r_m} + b_1) \times \sigma(\boldsymbol{r_t} \cdot \boldsymbol{r_h}), \text{ otherwise} \end{cases} \tag{15}$$

where W_1 and b_1 are parameters.

Asymmetric axiom consistency focuses on whether r is asymmetric, namely, whether (h, r, t) and (t, r, h) both exist. If r is asymmetric, and (h, r, t) and (t, r, h) appear simultaneously, an asymmetric inconsistency occurs. It applies a feed-forward layer to determine the property of the relation as similar to that defined in the irreflexive axiom. Meanwhile, we use TransE to check whether (t, r, h) exists. The asymmetric axiom network can be defined as follows:

$$f_{kge} = \sigma(\boldsymbol{t_m} + \boldsymbol{r_m} - \boldsymbol{h_m}), \ P_{asy} = \sigma(W_6(r_m) + b_6) \times f_{kge}. \tag{16}$$

where σ denotes sigmoid function.

4.2.3. Fusion

The overall score for each triple is defined as:

$$E(h, r, t) = E_S + E_{DM} + E_{RG} + E_{DIS} + E_{IRRE} + E_{ASYM}. \tag{17}$$

where E_S is the score of the KG embedding module, and $E_{DM} = \|1 - P_{dm}\|$, $E_{RG} = \|1 - P_{rg}\|$, $E_{DIS} = \|1 - P_{dis}\|$, $E_{IRRE} = \|1 - P_{irre}\|$, and

Table 2
Evaluation of event classification with overall instances. $P(\%)$, $R(\%)$ and $F(\%)$ stand for precision, recall, and F1-score respectively.

Model	P	R	F
DMCNN [17]	62.51 ± 1.10	62.35 ± 1.12	63.72 ± 0.99
JRNN [27]	63.73 ± 0.98	63.54 ± 1.13	66.95 ± 1.03
JMEE [40]	52.02 ± 1.14	53.80 ± 1.15	68.07 ± 1.02
AD-DMBERT [41]	67.35 ± 1.01	**73.46 ± 1.12**	71.89 ± 1.03
OneIE [42]	71.94 ± 1.03	68.52 ± 1.05	71.77 ± 1.01
PathLM [43]	73.51 ± 0.99	68.74 ± 1.03	72.83 ± 1.01
NSKE	**75.46 ± 1.06**	70.38 ± 1.12	**74.92 ± 1.07**

$E_{ASYM} = \|1 - P_{asym}\|$ are scores of domain, range, disjoint, irreflexive, asymmetric axioms. The lower value of E_S, the more likely the triple is to be correct. The higher value of P_{dm}, P_{rg}, P_{dis}, P_{irre}, and P_{asym}, the triple is more likely to satisfy this axiom.

We utilize a margin-based score function, which is defined as follows:

$$\mathcal{L}_t = \sum_{(h,r,t)\in T} \sum_{(h',r',t')\in T'} max(0, E(h,r,t) + \gamma - E(h',r',t')) \tag{18}$$

where γ denotes the margin. T is the positive triple set, and T' is the negative triple set which is denoted by:

$$T' = \{(h',r,t)|h' \in \mathcal{E} \cup (h,r,t')|t' \in \mathcal{E} \cup (h,r',t)|r \in \mathcal{R}\},\ (h,r,t) \in T \tag{19}$$

We use the binary cross-entropy loss for probabilities of axioms, which is defined by:

$$\mathcal{L} = \mathcal{L}_t + \mathcal{L}_{dm} + \mathcal{L}_{rg} + \mathcal{L}_{dis} + \mathcal{L}_{irr} + \mathcal{L}_{asy} \tag{20}$$

where $\mathcal{L}_t, \mathcal{L}_{dm}, \mathcal{L}_{rg}, \mathcal{L}_{dis}, \mathcal{L}_{irr}, \mathcal{L}_{asy}$ denotes the loss for domain, range, disjoint, irreflexive, asymmetric axioms.

5. Evaluation

5.1. Knowledge extraction

5.1.1. Experiment setting

We verify effectiveness of *NSKE* in two types of evaluation: (1) *Overall Evaluation*, and (2) *Low-resource Evaluation*.

Datasets. We use a newly proposed RE dataset [38] considering rules of relations, which contains 353 relations and 856,217 instances. In addition, we utilize a newly proposed EE dataset `OntoEvent` [39] annotated with event correlations. It contains 100 event types, and 3804 event-event relations, derived from 60,546 instances.

Baselines. For RE, we select DeViSE [28] and ConSE [29] as semantic project functions, integrated with three types of embedding representations and their pair-wise combinations, including *Word Embedding* E_{wd}, *KG Embedding* E_{kg}, *Rule-guided Embedding* E_{rl}, *KG + Word Embedding* E_{kw}, *Rule + Word Embedding* E_{rw}, *KG + Rule Embedding* E_{kr}.

For EE, we adopt CNN-based model DMCNN [17], RNN-based model JRNN [27], and GCN-based model JMEE [40]. In addition, we adopted the BERT-based model AD-DMBERT [41] with adversarial imitation learning. We also adopt the graph-based models OneIE [42] and PathLM [43], which generate graphs from event instances for EE.

5.1.2. Results and discussion

Overall Evaluation. We demonstrate the effectiveness of neural symbolic reasoning in standard KE through an overall evaluation of EE. We follow the evaluation protocol of standard EE models, *e.g.*, DMCNN [17]. Event instances are split into training, validating, and testing subsets with ratios of 0.8, 0.1, and 0.1, respectively. Note that there are no new event types in the testing set that are not seen in the training set. As shown in Table 2, NSKE achieves larger gains compared with the conventional EE baselines, *e.g.*, DMCNN, JRNN, and JMEE. Moreover, NSKE still generally outperforms the BERT-based model, AD-DMBERT. This implies the effectiveness of the KE framework with neural symbolic reasoning, which can leverage and propagate correlations among event types, thereby reducing the dependence on data to some extent. In particular, NSKE outperformed graph-based models, *i.e.*, OneIE, and PathLM. A possible reason is that although they both convert sentences into instance graphs, and PathLM even connects event types with multiple entities, the event correlations are still implicit and difficult to capture, whereas NSKE can explicitly utilize event correlations and directly propagate information among event types.

Low-resource Evaluation. We further demonstrate the effectiveness of neural symbolic reasoning in low-resource knowledge extraction by low-resource evaluation of RE, *i.e.*, zero-shot RE. Case studies for zero-shot REs over different types of semantic representations are listed in Table 3. This shows that most relations based on rule embedding achieve at least comparable results with KG embedding, such as *nominated_for, producer*, and *lyrics_by*. Some relations, such as *producer*, slightly outperform KG embedding. Maybe this is because rule embedding can capture logic-level connections between seen and unseen relations. For example, the unseen relation *nominated_for* is logically related to two seen relations *award_received* and *winner* with the rule *nominated_for(x,z)* $\Leftarrow$ *award_received(x,y)* $\wedge$ *winner(y,z)*. The most interesting aspect is the relation *mother* for which KG embedding fails to compare with word embedding because it is poorly trained, whereas rule embedding achieves comparable scores with word embedding. The reason may be that rule embedding helps strengthen the embedding by incorporating more knowledge from related relations contained in the rules, thus correcting the relation embedding.

5.2. Relational reasoning

5.2.1. Experiment setting

Datasets. We propose two zero-shot property link prediction datasets: a synthetic KG *Synthetic-ZS*, and a biomedical KG *UMLS-ZS*. *Synthetic-ZS* contain 550 entities, 51 properties, and 130 axioms. *UMLS-ZS* contain 135 entities, 46 properties, and 560 axioms. We simulate inference of the reasoner based on several axioms, which generate an answer entity set $\mathcal{A}$ with entities that can be directly inferred by axioms, a candidate entity set C coming from *Domain* and *Range* axioms, and another entity set $\mathcal{O}$ in which $\mathcal{A} \cap C = \mathcal{A} \cap \mathcal{O} = C \cap \mathcal{O} = \emptyset$. We randomly generate a rank number $r \in [1, |\mathcal{A}|]$ if the correct entity e is in $\mathcal{A}$, $r \in [|\mathcal{A}|, |\mathcal{A}| + |\mathcal{C}|]$ if $e \in C$ and $r \in [|\mathcal{A}| + |\mathcal{C}|, |\mathcal{A}| + |\mathcal{C}| + |\mathcal{O}|]$ otherwise.

Baselines. We re-imply R-GCN [32], DistMult [31], and ComplEx [13] with comparable performance on benchmark datasets. We do not compare rule mining methods with our approach because they fail to handle zero-shot problems.

5.2.2. Results and discussion

KG with zero-shot properties. We explore the benefit of neural axiomatic reasoning, mainly compared with methods that make predictions based only on facts. We use the zero-shot-property link prediction as an example. The experimental setup included a dataset with properties in the test and a valid dataset unseen in the training dataset.

In Table 4, these results illustrate that the automatic utilization of neighbor axioms significantly helps tripletGNN tackle unseen properties. These improvements could also be credited to the access of neighboring fact triplets. To further explore the contribution of incorporating axioms, we perform an ablation study, experimenting with tripletGNN on triplet graph with axiom triplet nodes removed. We implement the experiment with the same parameters for tripletGNN as used before removing both datasets. The ablation study results are marked with *(-axioms)* in Table 4. The results show that the performance of tripleGNN significantly decreases once axioms are removed, especially for UMLS-ZS dataset with a 79.1% decrease in the MRR for tripletGNN-DistMult and 64.6% for tripletGNN-ComplEx. This directly proves that the performance gain of tripleGNN on zero-shot-property link prediction is mainly attributed to axioms.

Table 3
Results of different embeddings on F1 score *w.r.t.*ConSE as project function, and related rules *w.r.t.*unseen relations.

Unseen Relations	F1-score						Related rules *w.r.t.*unseen relations
	$+E_{wd}$	$+E_{kg}$	$+E_{rl}$	$+E_{kw}$	$+E_{rw}$	$+E_{kr}$	
mother	**0.83**	0.40	0.77	0.53	0.80	0.78	*mother(x,z) ⇐ spouse(x,y) ∧ father(y,z)*
							mother(x,y) ⇐ child(y,x)
lyrics_by	0.06	0.52	0.51	0.49	0.48	**0.52**	*lyrics_by(x,y) ⇐ composer(x,y)*
nominated_for	0.56	**0.97**	0.96	**0.97**	0.96	0.96	*nominated_for(x,z) ⇐ award_received(x,y) ∧ winner(y,z)*
producer	0.41	0.52	**0.55**	0.54	0.52	0.53	*producer(x,y) ⇐ director(x,y)*
							producer(x,y) ⇐ screenwriter(x,y)
							producer(x,y) ⇐ cast_member(x,y)
field_of_work	0.04	0.14	0.29	0.11	0.29	**0.37**	*field_of_work(x,y) ⇐ occupation(x,y)*
connecting_line	0.00	0.10	0.43	0.28	0.42	**0.47**	*connecting_line(x,z) ⇐ adjacent_station(y,x) ∧ part of(y,z)*
residence	0.01	0.32	0.30	0.30	0.38	**0.39**	*residence(x,y) ⇐ place_of_birth(x,y)*
							residence(x,y) ⇐ place_of_death(x,y)

Table 4
Results for zero-shot-property link prediction. Bold numbers: the best results, underlined numbers: the better ones between tripletGNN and its corresponding KG embedding method.

	Synthetic-ZS				UMLS-ZS			
	MRR	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10
Reasoner	**0.668**	**0.606**	**0.721**	0.727	0.130	0.042	0.135	0.255
R-GCN	0.455	0.393	0.539	0.557	0.270	0.197	0.299	0.374
DistMult	0.466	0.438	0.475	0.514	0.109	0.044	0.101	0.235
ComplEx	0.385	0.336	0.400	0.476	0.100	.032	.100	.210
tripletGNN-DistMult	0.636	0.562	0.690	**0.769**	0.316	.202	**.364**	.490
tripletGNN-ComplEx	0.621	0.564	0.656	0.717	**0.336**	**0.230**	0.358	**0.547**
tripletGNN-DistMult*(-axioms)*	0.504	0.478	0.514	0.555	0.066	0.008	0.029	0.179
tripletGNN-ComplEx*(-axioms)*	0.540	0.489	0.562	0.636	0.119	0.040	0.109	0.275

Table 5
Noise detection results on noisy datasets based on WN18RR.

	WN18RR-10%	WN18RR-20%	WN18RR-40%
TransE	0.8428	0.8202	0.7961
CKRL(TransE)	0.8463	0.8234	0.7976
K-NAN(TransE)	**0.8738**	**0.8582**	**0.8463**

5.3. *Inconsistency checking*

5.3.1. *Experiment setting*

Datasets. We use WN18RR [44], which is a subset of WordNet, to assess the methodologies presented. Because there are no explicitly labeled noisy triples, we corrupt either s or o. Three noisy datasets with negative triples of 10%, 20%, and 40% of positive triples were generated. The negative triples will be imbued to the training set of the original dataset and be regarded as positive triples for training.

Baselines. We select TransE [8] and CKRL [45] as baselines, because both our model and CKRL are based on TransE. Moreover, it is not difficult for our proposed confidence-aware framework to be utilized in other enhanced translation-based methods, such as TransR [46].

5.3.2. *Results and discussion*

We explore the capability of our method to detect noise in a noisy KG. Inspired by the evaluation metric of noise detection in CKRL, we use $E(h,r,t) = \|\boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t}\|$ to calculate the triple scores. Triples with higher scores in the training set are considered as noise.

In Table 5, we can observe that: (1) Compared with TransE and CKRL, K-NAN achieves better results on all three datasets with different noise proportions. This confirms the efficiency of our method and demonstrates that axiom information can benefit noise detection. (2) As the noise rate increases, the performance of all the models deteriorates. However, K-NAN is more robust, as showing a much smaller influence on performance. (3) The results of the CKRL are not significantly improved compared to TransE, it may be caused by fewer relations. While K-NAN is more comparable, which implies that axiom information may be more useful than path information.

6. Related work

6.1. *Knowledge graph reasoning*

Knowledge reasoning over KGs aims to identify errors and infer new conclusions from existing data [47–50]. Two of the most common learning methods for KGR are embedding-based reasoning and rule-based reasoning [51]. Embedding-based methods such as TransE [8], HolE [12], and ComplEx [13] learn latent representations of entities and relations in continuous vector spaces. Embedding-based reasoning is more efficient when there are a large number of relations or triples to reason over. However, most embedding-based methods cannot model multi-hop relationship paths. In addition, their predictions were almost unexplainable. Rule-based methods, such as AMIE [11], aim to learn deductive and interpretable inference rules. Rule-based reasoning is precise and can provide insights into the inference results. However, rule-based methods cannot tolerate ambiguous data. In addition, the predefined evaluation indicators and the formation of rules limit the expression ability of the learned rules.

6.2. *Neural symbolic AI*

Over the past few decades, some studies have focused on symbolic inference [52]. Although symbolic reasoning is good at logical reasoning and has strong interpretability, it is difficult to deal with the uncertainty of entities, relationships, and ambiguity. In contrast, neural networks are fault-tolerant and further compare these embeddings through symbolic representation, rather than the literal meaning between entities and relations. The latest reasoning models combine these two reasoning methods, which generally contain three types. The first is for neural reasoning, which uses logic rules to improve the embedding in neural reasoning [53,54]. The second involves replacing neural reasoning with a probability framework, and establishing a probability model to infer the answer, thereby designing logical rules [11,55]. The third category aims at reasoning rules through symbolic reasoning, but combines neural networks to deal with data uncertainty and ambiguity [14,56], with the search space reduced in symbolic reasoning.

7. Conclusions

In this study, we propose a reasoning method for the entire lifecycle of KGs using neural symbolic methods. Compared with traditional symbolic and neural methods, neural symbolic reasoning with KGs integrates the advantages of these two approaches and considers both scalability and interpretability. Furthermore, we implement neural symbolic reasoning over three phases of the KG lifecycle, including extraction, completion, and refinement, which correspond to three subtasks: knowledge extraction, relational reasoning, and inconsistency checking. Experiments on the three subtasks demonstrate the effectiveness of our proposed neural symbolic methods. In the future, we intend to continuously investigate other neural symbolic KG reasoning methods and demonstrate their effectiveness in more downstream tasks.

Declaration of Competing Interest

The authors declare that they have no conflict of interest in this work.

Acknowledgments

We want to express gratitude to the anonymous reviewers for their hard work and kind comments. This work is funded by National Natural Science Foundation of China (Grant no. 91846204 and U19B2027) and National Key Research and Development Program of China (Grant no. 2018YFB1402800).

References

[1] K.D. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: SIGMOD Conference, ACM, 2008, pp. 1247–1250.
[2] D. Vrandecic, Wikidata: a new platform for collaborative data collection, in: WWW (Companion Volume), ACM, 2012, pp. 1063–1064.
[3] F. Li, H. Chen, G. Xu, T. Qiu, F. Ji, J. Zhang, H. Chen, Alimekg: domain knowledge graph construction and application in e-commerce, in: CIKM, ACM, 2020, pp. 2581–2588.
[4] W. Cui, Y. Xiao, H. Wang, Y. Song, S. Hwang, W. Wang, KBQA: learning question answering over QA corpora and knowledge bases, Proc. VLDB Endow. 10 (5) (2017) 565–576.
[5] M.E. Peters, M. Neumann, R.L.L. IV, R. Schwartz, V. Joshi, S. Singh, N.A. Smith, Knowledge enhanced contextual word representations, in: EMNLP/IJCNLP (1), Association for Computational Linguistics, 2019, pp. 43–54.
[6] L. Sang, M. Xu, S. Qian, X. Wu, Knowledge graph enhanced neural collaborative recommendation, Expert Syst. Appl. 164 (2021) 113992.
[7] J. Martínez-Rodríguez, I. López-Arévalo, A.B. Ríos-Alvarado, Openie-based approach for knowledge graph construction from text, Expert Syst. Appl. 113 (2018) 339–355.
[8] A. Bordes, N. Usunier, A. García-Durán, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: NIPS, 2013, pp. 2787–2795.
[9] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur, Y. Katz, Pellet: a practical OWL-DL reasoner, J. Web Semant. 5 (2) (2007) 51–53.
[10] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R.H. Jr., T.M. Mitchell, Toward an architecture for never-ending language learning, AAAI, AAAI Press, 2010.
[11] L.A. Galárraga, C. Teflioudi, K. Hose, F.M. Suchanek, AMIE: association rule mining under incomplete evidence in ontological knowledge bases, in: WWW, International World Wide Web Conferences Steering Committee / ACM, 2013, pp. 413–422.
[12] M. Nickel, L. Rosasco, T.A. Poggio, Holographic embeddings of knowledge graphs, in: AAAI, AAAI Press, 2016, pp. 1955–1961.
[13] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: ICML, in: JMLR Workshop and Conference Proceedings, 48, JMLR.org, 2016, pp. 2071–2080.
[14] F. Yang, Z. Yang, W.W. Cohen, Differentiable learning of logical rules for knowledge base reasoning, in: NIPS, 2017, pp. 2319–2328.
[15] W. Zhang, B. Paudel, L. Wang, J. Chen, H. Zhu, W. Zhang, A. Bernstein, H. Chen, Iteratively learning embeddings and rules for knowledge graph reasoning, in: WWW, ACM, 2019, pp. 2366–2377.
[16] J. Hao, M. Chen, W. Yu, Y. Sun, W. Wang, Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts, in: KDD, ACM, 2019, pp. 1709–1719.
[17] Y. Chen, L. Xu, K. Liu, D. Zeng, J. Zhao, Event extraction via dynamic multi-pooling convolutional neural networks, in: ACL (1), The Association for Computer Linguistics, 2015, pp. 167–176.
[18] N. Zhang, S. Deng, Z. Sun, J. Chen, W. Zhang, H. Chen, Relation adversarial network for low resource knowledge graph completion, in: WWW, ACM / IW3C2, 2020, pp. 1–12.
[19] S. Deng, N. Zhang, J. Kang, Y. Zhang, W. Zhang, H. Chen, Meta-learning with dynamic-memory-based prototypical network for few-shot event detection, in: WSDM, ACM, 2020, pp. 151–159.
[20] J. Liu, Y. Chen, K. Liu, W. Bi, X. Liu, Event extraction as machine reading comprehension, in: EMNLP (1), Association for Computational Linguistics, 2020, pp. 1641–1651.
[21] H. Ye, N. Zhang, S. Deng, M. Chen, C. Tan, F. Huang, H. Chen, Contrastive triple extraction with generative transformer, in: AAAI, AAAI Press, 2021, pp. 14257–14265.
[22] N. Zhang, X. Chen, X. Xie, S. Deng, C. Tan, M. Chen, F. Huang, L. Si, H. Chen, Document-level relation extraction as semantic segmentation, in: IJCAI, ijcai.org, 2021, pp. 3999–4006.
[23] D. Lou, Z. Liao, S. Deng, N. Zhang, H. Chen, Mlbinet: a cross-sentence collective event detection network, in: ACL/IJCNLP (1), Association for Computational Linguistics, 2021, pp. 4829–4839.
[24] Y. Lu, H. Lin, J. Xu, X. Han, J. Tang, A. Li, L. Sun, M. Liao, S. Chen, Text2event: Controllable sequence-to-structure generation for end-to-end event extraction, in: ACL/IJCNLP (1), Association for Computational Linguistics, 2021, pp. 2795–2806.
[25] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: NAACL-HLT (1), Association for Computational Linguistics, 2019, pp. 4171–4186.
[26] D. Zeng, K. Liu, Y. Chen, J. Zhao, Distant supervision for relation extraction via piecewise convolutional neural networks, in: EMNLP, The Association for Computational Linguistics, 2015, pp. 1753–1762.
[27] T.H. Nguyen, K. Cho, R. Grishman, Joint event extraction via recurrent neural networks, in: HLT-NAACL, The Association for Computational Linguistics, 2016, pp. 300–309.
[28] A. Frome, G.S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, T. Mikolov, Devise: A deep visual-semantic embedding model, in: NIPS, 2013, pp. 2121–2129.
[29] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. Corrado, J. Dean, Zero-shot learning by convex combination of semantic embeddings, ICLR, 2014.
[30] Q. Ning, Z. Feng, H. Wu, D. Roth, Joint reasoning for temporal and causal relations, in: ACL (1), Association for Computational Linguistics, 2018, pp. 2278–2288.
[31] B. Yang, W. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, ICLR (Poster), 2015.
[32] M.S. Schlichtkrull, T.N. Kipf, P. Bloem, R. van den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: ESWC, in: Lecture Notes in Computer Science, 10843, Springer, 2018, pp. 593–607.
[33] H. Paulheim, C. Bizer, Improving the quality of linked data using statistical distributions, Int. J. Semantic Web Inf. Syst. 10 (2) (2014) 63–86.
[34] S. Jang, Megawati, J. Choi, M.Y. Yi, Semi-automatic quality assessment of linked data without requiring ontology, in: NLP-DBPEDIA@ISWC, in: CEUR Workshop Proceedings, 1581, CEUR-WS.org, 2015, pp. 45–55.
[35] H. Paulheim, H. Stuckenschmidt, Fast approximate a-box consistency checking using machine learning, in: ESWC, in: Lecture Notes in Computer Science, 9678, Springer, 2016, pp. 135–150.
[36] C. Meilicke, D. Ruffinelli, A. Nolle, H. Paulheim, H. Stuckenschmidt, Fast abox consistency checking using incomplete reasoning and caching, in: RuleML + RR, in: Lecture Notes in Computer Science, 10364, Springer, 2017, pp. 168–183.
[37] T. Tran, M.H. Gad-Elrab, D. Stepanova, E. Kharlamov, J. Strötgen, Fast computation of explanations for inconsistency in large-scale knowledge graphs, in: WWW, ACM / IW3C2, 2020, pp. 2613–2619.
[38] J. Li, R. Wang, N. Zhang, W. Zhang, F. Yang, H. Chen, Logic-guided semantic representation learning for zero-shot relation classification, in: COLING, International Committee on Computational Linguistics, 2020, pp. 2967–2978.
[39] S. Deng, N. Zhang, L. Li, C. Hui, H. Tou, M. Chen, F. Huang, H. Chen, Ontoed: Low-resource event detection with ontology embedding, in: ACL/IJCNLP (1), Association for Computational Linguistics, 2021, pp. 2828–2839.
[40] X. Liu, Z. Luo, H. Huang, Jointly multiple events extraction via attention-based graph information aggregation, in: EMNLP, Association for Computational Linguistics, 2018, pp. 1247–1256.
[41] X. Wang, X. Han, Z. Liu, M. Sun, P. Li, Adversarial training for weakly supervised event detection, in: NAACL-HLT (1), Association for Computational Linguistics, 2019, pp. 998–1008.
[42] Y. Lin, H. Ji, F. Huang, L. Wu, A joint neural model for information extraction with global features, in: ACL, Association for Computational Linguistics, 2020, pp. 7999–8009.
[43] M. Li, Q. Zeng, Y. Lin, K. Cho, H. Ji, J. May, N. Chambers, C.R. Voss, Connecting the dots: Event graph schema induction with path language modeling, in: EMNLP (1), Association for Computational Linguistics, 2020, pp. 684–695.
[44] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: AAAI, AAAI Press, 2018, pp. 1811–1818.
[45] R. Xie, Z. Liu, F. Lin, L. Lin, Does william shakespeare REALLY write hamlet? knowledge representation learning with confidence, in: AAAI, AAAI Press, 2018, pp. 4954–4961.
[46] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: AAAI, AAAI Press, 2015, pp. 2181–2187.
[47] J. Chen, F. Lécué, J.Z. Pan, S. Deng, H. Chen, Knowledge graph embeddings for dealing with concept drift in machine learning, J. Web Semant. 67 (2021) 100625.
[48] W. Zhang, S. Deng, H. Wang, Q. Chen, W. Zhang, H. Chen, Xtranse: explainable knowledge graph embedding for link prediction with lifestyles in e-commerce, in: JIST (2), in: Communications in Computer and Information Science, 1157, Springer, 2019, pp. 78–87.
[49] S. Deng, N. Zhang, W. Zhang, J. Chen, J.Z. Pan, H. Chen, Knowledge-driven stock

trend prediction and explanation via temporal convolutional network, in: WWW (Companion Volume), ACM, 2019, pp. 678–685.

[50] N. Zhang, S. Deng, H. Chen, X. Chen, J. Chen, X. Li, Y. Zhang, Structured knowledge base as prior knowledge to improve urban data analysis, ISPRS Int. J. Geo Inf. 7 (7) (2018) 264.

[51] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, Proc. IEEE 104 (1) (2015) 11–33.

[52] L.G. Valiant, Knowledge infusion: In pursuit of robustness in artificial intelligence, in: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2008.

[53] S. Guo, Q. Wang, L. Wang, B. Wang, L. Guo, Jointly embedding knowledge graphs and logical rules, in: Proceedings of the 2016 conference on empirical methods in natural language processing, 2016, pp. 192–202.

[54] S. Guo, Q. Wang, L. Wang, B. Wang, L. Guo, Knowledge graph embedding with iterative guidance from soft rules, in: Proceedings of the AAAI Conference on Artificial Intelligence, 32, 2018.

[55] P.G. Omran, K. Wang, Z. Wang, Scalable rule learning via learning representation., in: IJCAI, 2018, pp. 2149–2155.

[56] W. Xiong, T. Hoang, W.Y. Wang, Deeppath: a reinforcement learning method for knowledge graph reasoning, in: EMNLP, Association for Computational Linguistics, 2017, pp. 564–573.

Huajun Chen is a full professor at the College of Computer Science and Technologies at Zhejiang University and serves as the Director of Joint Lab on Knowledge Engine of AZFT (Alibaba-Zhejiang University Joint Research Institute of Frontier Technology), and a deputy director of the Key Lab of Big Data Intelligence in Zhejiang Province. He received his bachelor's degree and Ph.D. from Zhejiang University in 2000 and 2004, respectively. He worked as a visiting assistant professor at Yale Center for Medical Informatics, Yale University (from June 2006 to June 2007), and a visiting scholar at the School of Computer Science of Carnegie Mellon University (from June 2007 to August 2008). His research interests include knowledge graph technologies, ontologies and the Semantic Web, information extraction and natural language processing, and applications in areas such as biomedicine, e-commerce, and smart cities.