Neural Axiom Network for Knowledge Graph Reasoning

Juan Li[§] [a], Wen Zhang [§] [b], Xiangnan Chen [a], Jiaoyan Chen [c] and Huajun Chen[*] [a]
[a] *College of Computer Science and Technology, Zhejiang University, 38 Zheda Rd, Hangzhou, China*
E-mails: lijuan18@zju.edu.cn, xnchen2020@zju.edu.cn, huajunsir@zju.edu.cn
[b] *School of Software Technology, Zhejiang University, 38 Zheda Rd, Hangzhou, China*
E-mail: wenzhang2015@zju.edu.cn
[c] *Department of Computer Science, University of Oxford, 15 Parks Rd, Oxford OX1 3QD, UK*
E-mail: jiaoyan.chen@cs.ox.ac.uk

Abstract. Knowledge graph reasoning is essential for improving the quality of knowledge graphs due to automatic mechanisms involved in KG construction which probably introduces incompleteness and incorrectness. In recent years, various KG reasoning techniques such as symbolic- and embedding-based methods, have been proposed for inferring missing triples and detecting noises. Symbolic-based reasoning methods concentrate on inferring new knowledge according to predefined rules or ontologies, where rules and axioms have been proved to be effective but are difficult to obtain. Meanwhile, embedding-based reasoning methods learn low-dimensional representations of entities and relations primarily by utilizing structural information, and the learned embeddings achieve promissing results in downstream tasks such as knowledge graph completion. These methods, however, ignore implicit axiom information which are not predefined in KGs but can be reflected through data. To be specific, each correct triple is considered to satisfy all axioms, as it is also a consistent triple. In this paper, we explore how to combine explicit structural and implicit axiom information to improve reasoning ability. Specifically, we present a novel **NeuR**al **A**xiom **N**etwork framework (**NeuRAN**) that only uses existing triples in KGs to address issues in the above methods. The framework consists of a knowledge graph embedding module that preserves the semantics of a triple, and five axiom modules that are encoded based on the characteristics of five kinds of axioms corresponding to five typical object property expression axioms defined in OWL2, including *ObjectPropertyDomain, ObjectPropertyRange, DisjointObjectProperties, IrreflexiveObjectProperty* and *AsymmetricObjectProperty*. The knowledge graph embedding module and axiom modules respectively calculate the probabilities that the triple conforms to the semantics and the corresponding axioms. Evaluations on KG reasoning tasks including noise detection, triple classification and link prediction show the efficiency of our method.

Keywords: Knowledge Graph Reasoning, Knowledge Graph Embedding, Noise Detection, Triple Classification, Link Prediction

1. Introduction

Knowledge Graphs (KGs) are represented as multi-relational directed graphs composed of entities as nodes and relations as edges, where knowledge is organized in the form of triples (*subject entity, relation, object entity*), abbreviated as (s, r, o). Typical KGs like DBpedia[1], Freebase[2], Wikidata[3], and Yago[4], have played a pivotal role in a broad range of applications, such as question answering[5] and recommender system[6]. Since KGs are usually automatically created and contain billions of triples, it is inevitable that they may suffer from incompleteness and incorrectness. For example, 71% of people in Freebase have no place of birth, and 94% have no known parents[7]. While Wikipedia is estimated to have 2.8% of its statements wrong[8]. To deal with these issues, methods proposed for knowledge graph reasoning tasks have received increasing attention. There are two mainstream techniques, including symbolic- and embedding-based methods.

[§]Equal contribution.
[*]Corresponding author.

Symbolic-based methods[9–12] adopt logic rules or ontologies for KG reasoning, and have shown to achieve good performance. For example, if an axiom *DisjointObjectProperties(:hasParent :hasSpouse)* indicating that the relations *hasSpouse* and *hasParent* are disjoint has been already defined, and *(Linda, hasSpouse, Bruce)* is a correct triple, the triple *(Linda, hasParent, Bruce)* will be classified as an incorrect triple. Although this kind of methods are more reliable and human-interpretable if used in reasoning tasks, they require rich ontologies which are usually missing or incomplete in KGs. Moreover, manual definition of axioms is tedious, and maintaining them is a non-trivial task.

Embedding-based methods[13–16] target to embed entities and relations into low-dimensional vector space. Based on these embeddings, a scoring function is employed to compute scores for triples to measure the plausibility of them. Knowledge graph embedding (KGE) models, such as translation-based methods[13, 17, 18], semantic-based methods[14, 19], and neural network methods[16, 20, 21], have been shown to be scalable and effective. However, despite the successes of embedding models in KG completion which aims to predict missing links, as well as in noise detection which aims to detect noisy triples. They focus on structural information rather than taking into consideration of implicit axiom information. Take implicit domain axiom as an example. Given the positive triple $(Linda, hasSpouse, Bruce)$, even if type of the subject entity *Linda* and domain of the relation $hasSpouse$ are not given, we can infer that it satisfies *domain* axiom.

In this paper, we propose a neural axiom network framework NeuRAN for KG reasoning. This framework not only encodes explicit structural information through a knowledge graph embedding model, but also implicit axiom information through neural networks. The main idea behind is that even ontology information is not explicitly defined in the given KG, any correct triple satisfies all axioms. In specific, we consider five different axioms corresponding to five typical object property expression axioms selected from OWL2 ontology language[*], including *ObjectPropertyDomain, ObjectPropertyRange, DisjointObjectProperties, IrreflexiveObjectProperty* and *AsymmetricObjectProperty*. As *domain* and *range* axioms are related to type compatibility, we distinguish type and semantic embeddings. Each entity has one type and one semantic vector representation. And each relation has two type (i.e., subject and object entity types excepted by the relation) and one semantic embeddings. We encode inherent structure of triples via an embedding module to learn semantic embeddings of entities and relations, such as TransE and TransH. Without predefined axioms, we introduce five axiom modules where semantic and type embeddings are involved, to calculate probability scores range from 0 to 1 for each possible type of axiom. The probability scores indicate the degree to which the triple satisfies the axioms. The closer the value is to 1, the more likely the triple is to satisfy the axiom. The design of these axiom modules are based on conditions satisfied by the axioms as listed in Table 1. For each triple (s, r, o), *domain/range* axiom module concentrates on type compatibility between subject/object entity type embedding expected by the relation and type embedding of the subject/object entity. For example in Figure 1, the triple *(United Kingdom, occupation, novelist)* violates *domain* axiom, thus the expected subject entity type vector of the relation *occupation* and type embedding of the subject entity *United Kingdom* may get a low *domain* axiom score close to 0. *Disjoint* axiom module focuses on compatibility of two relations with the same subject and object entity. *Irreflexive* axiom module encodes whether the relation is irreflexive and whether $s = o$. And *asymmetric* axiom module encodes whether the relation is asymmetric and whether (o, r, s) is also in the KG. The final score of the triple is defined as the summarization of scores from the structure and the five implicit axioms.

In summary, our main contributions are as follows:

- We raise the problem of neural axiom learning, in which axiom information is not given but can be reflected by and learned from existing data.
- We propose the framework NeuRAN with the consideration of both explicit structural information and implicit axiom information in KGs, relying exclusively on the triples in KGs for KGR tasks. The explicit structural and implicit axiom information is respectively encoded by a KG embedding module and five axiom modules.
- We evaluate NeuRAN on datasets with different ratio of noises and achieve promising performances, demonstrating the effectiveness of our models on noisy KGs.

[*]https://www.w3.org/TR/owl2-primer/

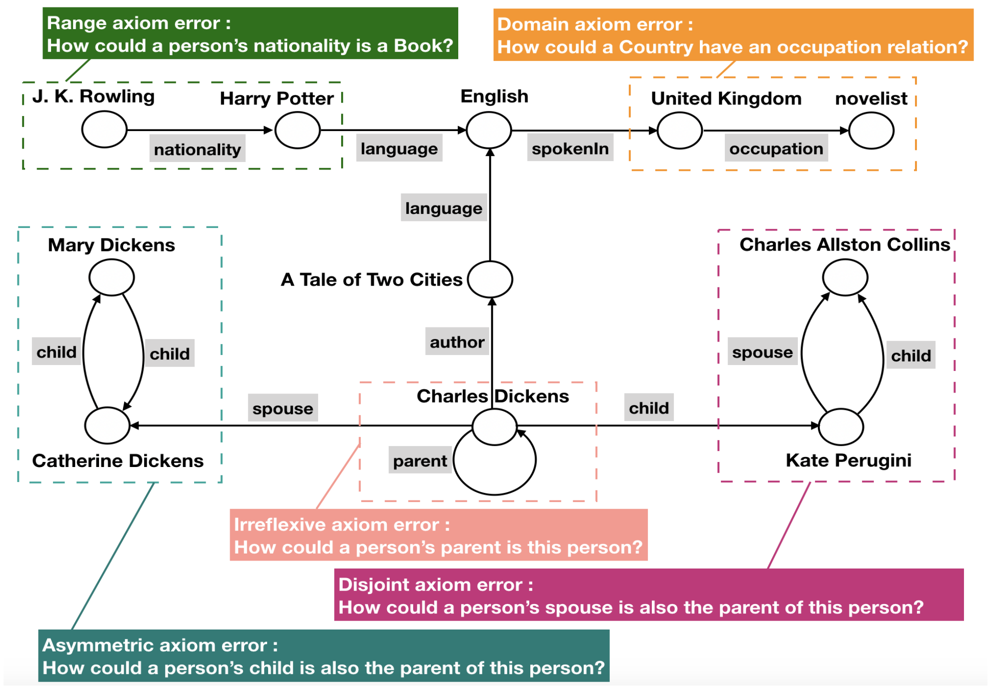

Fig. 1. In the hypothetical knowledge graph, there may exist erroneous triples. And the reason for these errors is that the triples do not conform to the axioms considered in this paper, including *domain*, *range*, *disjoint*, *irreflexive* and *asymmetric* axioms.

2. Related work

We discuss the following three lines of research work that are closely relevant to this paper.

2.1. Symbolic-based Reasoning

Symbolic-based reasoning is important to ensure the quality of KGs, aiming at inferring new knowledge or detecting noises with respect to accompanying rules or ontologies. Since rules and axioms suffer from incompleteness and manually annotate such information is infeasible, there are several works which attempt to enrich ontologies. For example, Inductive logic programming (ILP) has been used to mine logical rules, but it has limitations due to open-world assumption of KGs. AMIE[9] and AMIE+[10] make up for this shortcoming by introducing an altered confidence metric based on the partial completeness assumption. With the rules generated with AMIE+, [22] discovers inverse and symmetric axioms by applying the predefined reasoning rules. Moreover, [23] presents approaches based on statistical inductive learning, including correlation computing and association rule mining to enrich ontologies with disjointness axioms. As it evaluates the validity of association rule mining by computing the precision and recall scores, another association rule mining algorithm [24] not only discusses the precision and recall, but also analyzes quality of disjoint axioms acquired. In addition to disjoint axiom, [25] enriches DBpedia ontology with domain and range restrictions as well as class disjointness axioms, and uses enhanced ontology for error detection.

2.2. Embedding-based Reasoning

Knowledge graph embedding embeds entities and relations of a KG into a continuous vector space to preserve the structure information of the KG. There are mainly three categories of embedding models: translational distance, semantic matching and neural network models. Translational distance models learn embeddings by translating a subject entity to an object entity through a relation. For example, TransE[13] represents entities and relations in the same vector space and assumes $(\boldsymbol{s} + \boldsymbol{r})$ to be close to $\boldsymbol{o}$, where $\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}$ are vector embeddings for s, r and o respectively. However, it has difficulty dealing with complex relations. To overcome the flaws, TransH[17] introduces relation-specific hyperplanes to allow entities have different embeddings in different relations. TransR[18] builds entity and relation embeddings in separate entity

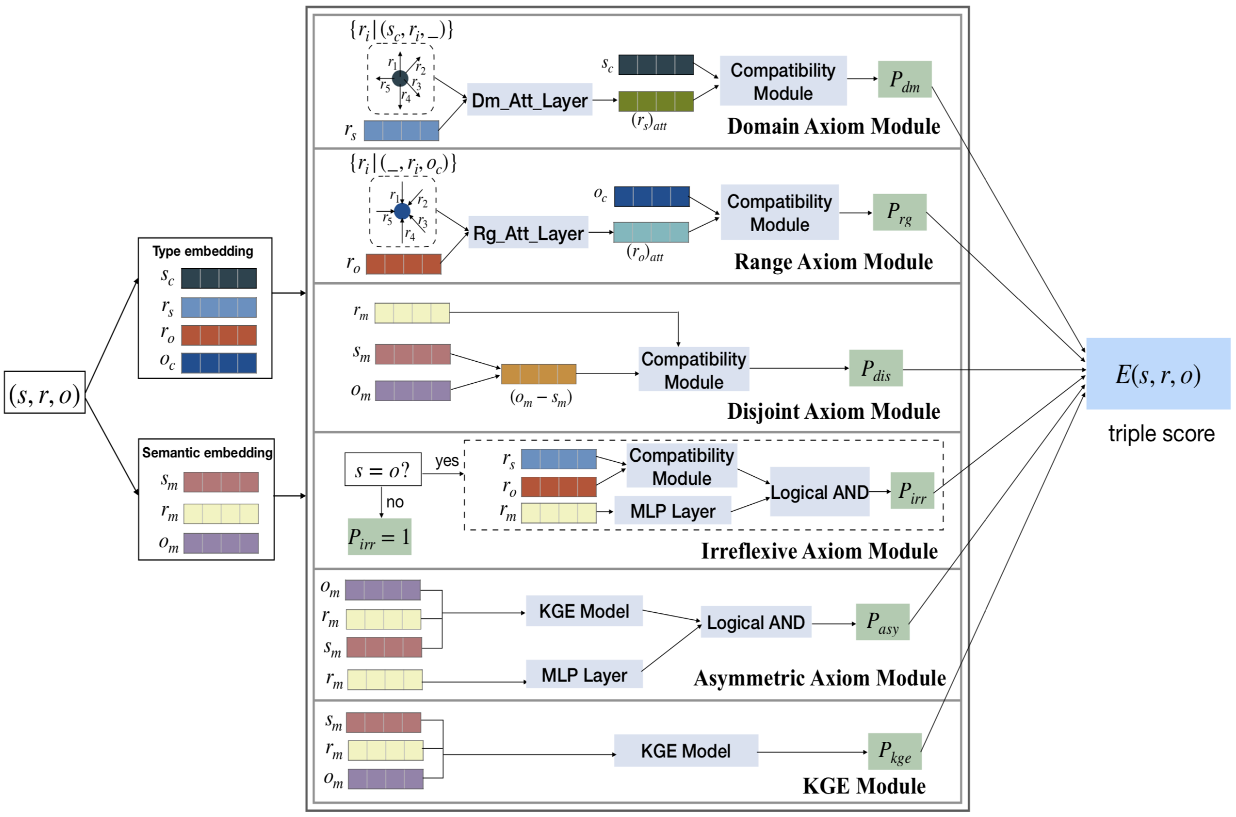

Fig. 2. The key idea of our framework.

and relation spaces. And TransD[26] constructs mapping matrices dynamically. Similarly, TorusE[27] and RotatE[28] use lie groups and rotations for translation respectively. Semantic matching models measure plausibility by matching latent semantics of entities and relations. DistMult[14] uses a formulation of bilinear model to represent entities and relations. ComplEx[19] extends DistMult by introducing complex-valued embeddings so as to better model asymmetric relations. HolE[15] makes use of circular correlation of embedding to learn compositional representations and semantically matches circular correlation with the relation embedding. Apart from that, researchers have raised interests in applying neural networks for knowledge graph reasoning. ConvE[16] and ConvKB[20] employ convolutional neural network to achieve better link prediction performance. CapsE[21] explores a capsule network to model relationship triples. Also, KGTtm[29] and CKRL[30] measure trustworthiness or confidence of triples.

2.3. Hybrid Reasoning

Another line of work concerns hybrid methods for KG reasoning, such as the combination of symbolic reasoning and embedding-based reasoning, and the combination of symbolic reasoning and statistical reasoning. In the former methods, TransC[31] learns SubClassOf axiom between types by encoding each type as a sphere and each entity as a vector. Furthermore, SetE[32] computes two axioms SubClassOf and SubPropertyOf in subsumption by employing linear programming methods on embeddings, focusing on domain, range or subClassOf axioms. Recently, IterE[33] iteratively learns embeddings and rules, considers seven object property expression axioms for rule learning. It combines rule learning and embedding learning to improve the quality of sparse entity embeddings by injecting new triples about sparse entities according to the scores of the axioms as well as to generate high quality rules. As for the latter, the statistic-based methods such as SDType and SDValidate[34], exploit statistical distributions of types and relations. SDType

Table 1

Five types of object property expression axioms selected from OWL2 ontology language. OP is the short for ObjectProperty. OPE denotes Object Property Expression, and x, y, z are entity variables. $\triangle_I$ is a nonempty set called the object domain. $\cdot^{OP}$ is an object property interpretation function. When translating axioms into examples in KG according to condition, we replace OPE in axioms with a relation.

Object Property Axioms	Condition	Examples
OPDomain(OPE CE)	$\forall(x,y) \in (OPE)^{OP}$ implies $x \in (CE)^C$	Domain(hasWife, Man)
OPRange(OPE CE)	$\forall(x,y) \in (OPE)^{OP}$ implies $y \in (CE)^C$	Range(hasWife, Woman)
DisjointOP(OPE$_1$...OPE$_n$)	$(OPE_j)^{OP} \cap (OPE_k)^{OP} = \varnothing$ for each $1 \leqslant j \leqslant n$ and each $1 \leqslant k \leqslant n$ such that $j \neq k$	Disjoint(hasParent, hasSpouse)
IrreflexiveOP(OPE)	$\forall x : x \in \triangle_I$ implies $(x,x) \notin (OPE)^{OP}$	Irreflexive(parentOf)
AsymmetricOP(OPE)	$\forall(x,y) \in (OPE)^{OP}$ implies $(y,x) \notin (OPE)^{OP}$	Asymmetric(hasChild)

deduces missing type information based on statistics about the usage of relations with entities of known type. And SDValidate measures the deviation between actual types of the subject and/or object and the apriori probabilities given by the distribution.

3. Method

We begin this section by briefly describing some notations. We denote a knowledge graph as $\mathcal{G} = \{(s,r,o) \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}$, where $s, o \in \mathcal{E}$, $r \in \mathcal{R}$, $\mathcal{E}$ is the entity set, and $\mathcal{R}$ is the relation set. (s,r,o) is a triple indicates the relation r between the subject entity s and the object entity o. Throughout this paper, we use bold letters to indicate vectors. $\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}$ are the embedding vectors of s, r, o respectively. And the abbreviations $DM(dm)$, $RG(rg)$, $DIS\,(dis)$, $IRRE(irre)$ and $ASYM(asym)$ respectively correspond to the implicit *domain*, *range*, *disjoint*, *irreflexive* and *asymmetric* axioms.

Then, we introduce the neural axiom network NeuRAN that combines a knowledge graph embedding module and five axiom modules(§3.1). Next, we introduce the KGE module which is used to encode explicit structural information (§3.2), and the five axiom modules which aim to encode five kinds of implicit axiom information (§3.3).

3.1. Neural Axiom Network

As depicted in Figure 2, we present a neural axiom network designed for KG reasoning, encoding both explicit structural(KGE module) and implicit axiom information(different axiom modules) existing in triples. In addition to the semantics of triples, NeuRAN takes into account of five kinds of axioms including *domain, range, disjoint, irreflexive* and *asymmetric* axioms implicit in each triple. We assume that probability value of the degree of satisfaction of the axioms intensify or mitigate the probability of existence of a triple. The score of a triple is calculated as:

$$E(s,r,o) = E_S + \alpha E_{NA} \tag{1}$$

$$E_{NA} = E_{DM} + E_{RG} + E_{DIS} + E_{IRRE} + E_{ASYM} \tag{2}$$

The overall energy function consists of two parts: E_S is the energy function which depends on the structure-based representations. It can be any of the knowledge graph embedding models[35]. A lower E_S indicates that the triple is more likely to be correct. The second part E_{NA} is the score of neural axioms, which consists of five kinds of axioms. $E_{DM} = ||1 - P_{dm}||$, $E_{RG} = ||1 - P_{rg}||$, $E_{DIS} = ||1 - P_{dis}||$, $E_{IRRE} = ||1 - P_{irre}||$, $E_{ASYM} = ||1 - P_{asym}||$ are respectively the energy functions of *domain, range, disjoint, irreflexive, asymmetric* axioms. P_{dm}, P_{rg}, P_{dis}, P_{irre} and P_{asym} correspond to probabilities that corresponding axioms are satisfied. The higher P_{dm}, P_{rg}, P_{dis}, P_{irre}, P_{asym}, and the lower E_{DM}, E_{RG}, E_{DIS}, E_{IRRE}, E_{ASYM} imply that the triple is more likely to satisfy the corresponding neural axiom.

Following the conventional training strategy of previous models, we train NeuRAN based on the local-closed world assumption. In this case, the observed triples in KGs are regard as positive triples, while the unobserved ones as negative triples. For the score of the triple, we utilize a margin-based score function, which is defined as follows:

$$\mathcal{L}_t = \sum_{(s,r,o)\in T} \sum_{(s',r',o')\in T'} max(0, E(s,r,o) + \gamma - E(s',r',o')) \tag{3}$$

where γ is a margin hyper-parameter, $E(s,r,o)$ is the overall energy function for (s,r,o). T and T' are the positive and negative triple sets. Since there are no explicit negative triples, T' can be generated by randomly corrupting the subject or object entity:

$$T' = \{(s', r, o) | s' \in \mathcal{E}\} \cup \{(s, r, o') | o' \in \mathcal{E}\}, \\ (s, r, o) \in T, (s', r, o) \notin T and (s, r, o') \notin T \tag{4}$$

In the process of generating negative triples, the goal is to minimize the loss function $\mathcal{L}_t$. And $\mathcal{L}_t$ is calculated by the embeddings of entities and relations.

Since structural information is encoded by a knowledge graph embedding model. In order to encode such latent axioms, we learn embeddings of entities and relations by adopting five axiom modules. Each possible type of axiom module is devised based on the condition of the axiom, the input of which is a triple, and the output is the probability of the triple conforming to the corresponding axiom. Considering *domain/range* axioms, as the two axioms are associated with type compatibility and type information is not provided, we attempt to introduce type embeddings. Following the type-sensitive models TypeDM and TypeComplex[36], we distinguish type embeddings from semantic embeddings when we learn vector representations for entities and relations. Thus, each entity is represented as two vectors, and each relation is represented as three vectors. Given a triple (s, r, o), for entities, we use type embeddings($\boldsymbol{s_c}$ and $\boldsymbol{o_c}$) and semantic embeddings($\boldsymbol{s_m}$ and $\boldsymbol{o_m}$) to represent the subject entity s and the object entity o respectively. As for the relation, $\boldsymbol{r_s}$ and $\boldsymbol{r_o}$ represent subject type and object type embeddings expected by the relation r, and $\boldsymbol{r_m}$ is the semantic embedding of r.

3.2. KG Embedding Module

The knowledge graph embedding module of the framework concerns the learning of a function E_S, which is designed to score each triple in KGs. Our framework considers two translation-based embedding models TransE[13] and TransH[17] as basic KG embedding models.

3.2.1. TransE

TransE is the simplest translation-based model, which interprets relations as translating operations between subject and object entities. The basic idea is that, a given triple (s, r, o) follows the assumption that $\boldsymbol{s} + \boldsymbol{r} \approx \boldsymbol{o}$ when (s, r, o) holds. The fitness of the model is calculated through the scoring function, which is defined as:

$$E_S = ||\boldsymbol{s_m} + \boldsymbol{r_m} - \boldsymbol{o_m}||_{L_1/L_2} \tag{5}$$

where L_1 and L_2 respectively denote the L_1 and L_2 norm. $\boldsymbol{s_m}$, $\boldsymbol{r_m}$ and $\boldsymbol{o_m}$ are the semantic embeddings of the subject entity, relation and object entity respectively. The smaller value of the scoring function, the higher the fitness for a triple.

3.2.2. TransH

TransH extends TransE by translating on hyperplanes, which models the relation $\boldsymbol{r}$ as a vector on a hyperplane with $\boldsymbol{w_r}$ as the normal vector. It enables an entity to have distinct representations when involved in different relations. Similar to TransE, the score function is defined as:

$$E_S = \|(\boldsymbol{s_m})_\perp + \boldsymbol{r_m} - (\boldsymbol{o_m})_\perp\|_{L_1/L_2} \tag{6}$$

where the projections $(\boldsymbol{s_m})_\perp = \boldsymbol{s_m} - \boldsymbol{w}_r^\top \boldsymbol{s_m} \boldsymbol{w}_r$, and $(\boldsymbol{o_m})_\perp = \boldsymbol{o_m} - \boldsymbol{w}_r^\top \boldsymbol{o_m} \boldsymbol{w}_r$. It restricts $\|\boldsymbol{w}_r\|_2 = 1$. The score is low if (s, r, o) holds, and is high otherwise.

3.3. Five Axiom Modules

Using only triple scores based on semantics for training ignores axiom information that is implicit but inherently existed in triples. It is intuitive that a correct triple, such as *(J. K. Rowling, nationality, United Kingdom)*, is also a logically consistent triple. Even without any priori knowledge of the ontological information, such as type information of the subject entity *J. K. Rowling* and the object entity *United Kingdom*, as well as *domain* and *range* of the relation *nationality*. We can infer that it satisfies *domain, range, disjoint, irreflexive*, and *asymmetric* axioms.

Therefore, for *domain/range* axioms, the type embedding of the subject/object entity expected by the relation $\boldsymbol{r_s}/\boldsymbol{r_o}$ and the type embedding of the subject/object entity $\boldsymbol{s_c}/\boldsymbol{o_c}$ are used to calculate a type compatibility score. For *disjoint* axiom, the semantic compatibility of any two relations with the same subject and object entities are discussed. We assume that $(\boldsymbol{o_m} - \boldsymbol{s_m})$ represent the general semantic embedding of the relation for the subject to be s and object to be o. For *irreflexive* axiom, whether the relation is irreflexive, and whether the subject entity is the same as the object entity($s = o$?) are considered. For *asymmetric* axiom whether the relation is asymmetric, and whether the symmetric triple of the input triple exists($(o, r, s) \in G$?) are concerned. We introduce these typical axioms in detail.

3.3.1. Domain Axiom Module

Domain axiom module focuses on type compatibility between the subject entity type expected by the relation r and type of the subject entity s. TypeDM uses the function $C(\boldsymbol{s_c}, \boldsymbol{r_s}) = \sigma(\boldsymbol{s_c} \cdot \boldsymbol{r_s})$ to measure the compatibility by calculating a score with the type embeddings of subject entity $\boldsymbol{s_c}$ and the subject entity type embedding expected by the relation $\boldsymbol{r_s}$. However, the subject type expected by the given relation may varies. For example, given the triple $(Soul(film), language, English)$, if we only focus on the relation *language*, the subject type expected by the relation can be an entity in the class of *Person, Book* or *Film*. But if we concern about the other relations of the subject entity, such as *starring* and *running time*, we may find that the subject entity expected by the relation is more likely to be in the class of *Film*. These relations may have strong correlations with the given relation, such as *starring* and *language*, or differ greatly such as *running time* and *language*. Thus, they contribute to embeddings of the subject entity type expected by the current given relation, and have different contributions. In order to represent embedding of the subject entity type expected by the given relation more accurately, we introduce attention mechanism to make use of rich information contained in the set of all the relations of the subject entity s which is $\mathcal{R}(s) = \{r_i | (s, r_i, e) \in \mathcal{G}\}$, where e denotes any entity in the KG. By adopting a domain attention layer (Dm_Att_Layer), we generate the subject type embedding expected by the relation $\boldsymbol{\hat{r}_s}$ based on the relations in $\mathcal{R}(s)$, a representation of each relation of the subject entity is learned. For each relation $r_i \in \mathcal{R}(s)$, we compute attention weights for relations of the subject. The importance is denoted by a_i, which reflects how relevant or important the relation r_i is to r_s.

$$a_i = f(\boldsymbol{r_s}, \boldsymbol{r_i}) = \boldsymbol{r_s}^T \boldsymbol{r_i}, r_i \in \mathcal{R}(s) \tag{7}$$

To get the relative attention values, *softmax* is applied over a_i.

$$p_i = \frac{exp(a_i)}{\sum_{r_j \in \mathcal{R}(s)} exp(a_j)} \tag{8}$$

where j denotes the jth relation of the subject entity. The new embedding of the subject entity type expected by the relation $\boldsymbol{\hat{r}_s}$ is the sum of the product of representation of each relation and the relation weighted by attention values of the considered relation.

$$\boldsymbol{\hat{r}_s} = \sum_{r_i \in \mathcal{R}(s)} p_i r_i \tag{9}$$

Then the type compatibility is calculated via a compatibility module, the likelihood of s and r satisfying *domain* axiom can be defined as follows:

$$P_{dm} = f(\boldsymbol{s_c}, \boldsymbol{r_s}) = \sigma(\boldsymbol{s_c} \cdot \boldsymbol{\hat{r}_s}) \tag{10}$$

where σ denotes sigmoid function.

3.3.2. Range Axiom Module

Range axiom module focuses on type compatibility between the object entity type expected by the relation r and the type of the object entity o. Similarly, TypeDM uses $C(\boldsymbol{o_c}, \boldsymbol{r_o}) = \sigma(\boldsymbol{o_c} \cdot \boldsymbol{r_o})$ to compute the compatibility score between the type embedding of object entity o_c and the object entity type embedding expected by the relation r_o, where the score can be used to evaluate the satisfaction of *range* axiom. However, a relation can have object entities with very different types. For example, the object entity of the relation *hasPart* can be *Leg* in (*Table, hasPart, Leg*), or *NewYorkBay* in (*Atlantics, hasPart, NewYorkBay*). Similar to the issue in *domain* axiom, object entities expected by a relation may exhibit diverse roles within the same relation, and other relations of the object entity may make different contributions to the embedding of the object entity type expected by the relation. To tackle this issue, we devise an attention mechanism as in *domain* axiom discerning the expected object entity type associated with the given relation. Taking the object entity o as a target, we denote the relations connected to o as $\mathcal{R}(o) = \{r_i | (e, r_i, o) \in \mathcal{G}\}$. By using a range attention layer (Rg_Att_Layer), we generate a new type embedding for the object entity expected by the relation $\boldsymbol{\hat{r}_o}$ based on all the relations in $\mathcal{R}(o)$. The importance of each relation to r denoted by b_i can be calculated as:

$$b_i = f(\boldsymbol{r_o}, \boldsymbol{r_i}) = \boldsymbol{r_o}^T \boldsymbol{r_i}, r_i \in \mathcal{R}(o) \tag{11}$$

We then apply *softmax* over b_i to get the relative attention values.

$$q_i = \frac{exp(b_i)}{\sum_{r_k \in \mathcal{R}(o)} exp(b_k)} \tag{12}$$

where k denotes the kth relation in the connected relations of the object entity. The generated embedding $\boldsymbol{\hat{r}_o}$

is the sum of the product of each object entity representation and the relation weighted by attention values of the object entities.

$$\hat{\boldsymbol{r_o}} = \sum_{r_i \in \mathcal{R}(o)} q_i r_i \tag{13}$$

The compatibility probability whether the triple satisfies *range* axiom is calculated by a compatibility module, and is defined as:

$$P_{rg} = f(\boldsymbol{o_c}, \boldsymbol{r_o}) = \sigma(\boldsymbol{o_c} \cdot \hat{\boldsymbol{r_o}}) \tag{14}$$

where σ denotes sigmoid function.

3.3.3. Disjoint Axiom Module

Disjoint axiom module focuses on the compatibility of the semantic embeddings of two relations with the same subject and object entities. For example, suppose *(John, spouse, Mary)* is an existing correct triple, then *(John, friend, Mary)* is correct, and *(John, child, Mary)* is incorrect. It is because that the two relations *spouse* and $friend$ can exist between two persons at the same time, while *spouse* and *child* are disjoint relations. In other words, if one person is the *spouse* of another, the two persons does not exist *children* relation, *spouse* and *children* could defined to be semantically *disjoint*. Following the condition of *disjoint* axiom, in order to find out disjoint triples for triple (s, r, o), we have to traverse the whole knowledge graph to find out all the relations $\mathcal{R}(s, o) = \{r_k | (s, r_k, o) \in \mathcal{G}, r_k \neq r\}$, and calculate the semantic compatibility of the relation pairs (r, r_k), which is time-consuming. To reduce time cost, we simply copy the idea from TransE, which holds the view that $\boldsymbol{s} + \boldsymbol{r} = \boldsymbol{o}$. Specifically, we attempt to regard $(\boldsymbol{o_m} - \boldsymbol{s_m})$ as the general representation of relations in triples with s and o being the subject and object entity respectively. Therefore, for relations $r_k \in \mathcal{R}(s, o)$, we assume they share similar semantic vector representation $(\boldsymbol{o_m} - \boldsymbol{s_m})$. The semantic judgment of this axiom can be simplified to calculate the compatibility score of $(\boldsymbol{o_m} - \boldsymbol{s_m})$ and $\boldsymbol{r_m}$, which is defined as:

$$f(\boldsymbol{r_m}, (\boldsymbol{o_m} - \boldsymbol{s_m})) = \sigma(\boldsymbol{r_m} \cdot (\boldsymbol{o_m} - \boldsymbol{s_m})) \tag{15}$$

where σ denotes sigmoid function.

3.3.4. Irreflexive Axiom Module

Irreflexive axiom module considers two aspects of judgements. One is the property of the relation *(i.e., whether r is irreflexive)*, and the other is whether the subject and object entity are equal*(i.e., whether s and o are the same entity)*. In OWL2, a relation is irreflexive means that no entity can be related to itself by such a relation. Therefore, if in the triple (s, r, o) the relation r is irreflexive and $s = o$, it violates *irreflexive* axiom. For example, when the relation is $hasParent$, it is intuitively that $(John, hasParent, John)$ is an incorrect triple.

Due to that we can judge whether $s = o$ directly without the need to represent the two entities as vectors, we conduct this as the first step. Only when $s = o$, we need to consider the property of the relation. Otherwise, the probability that the triple conforms to *irreflexive* axiom is 1, and the reason for the triple to be classified as wrong can not be due to the *irreflexive* axiom. If $s = o$, we can infer that types of s and o are the same ($\boldsymbol{s_c} = \boldsymbol{o_c}$). In regard to the property of the relation, we expect types of the subject entity and the object entity expected by the relation which are r_s and r_o should be respectively compatible with s and o. If so, it can be conclude that $\boldsymbol{r_s} \approx \boldsymbol{s_c}$ and $\boldsymbol{r_o} \approx \boldsymbol{o_c}$. As a result, r_s and r_o are compatible ($\boldsymbol{r_s} \approx \boldsymbol{r_o}$), which is measured by a compatibility module. The type constraint is calculated as $\sigma(\boldsymbol{r_o} \cdot \boldsymbol{r_s})$. Besides, the semantic information of the relation $\boldsymbol{r_m}$ can also help to determine the property of the relation. We utilize a multi-layer perceptron (MLP) layer to encode semantic information. Since violation of this axiom is determined by both types and semantics, we calculated the probability that a triple satisfies *irreflexive* axiom through the Logic AND operation. The final probability is defined as:

$$P_{irr} = \begin{cases} 1, if\ s \neq o \\ \sigma(W_1(\boldsymbol{r_m}) + b_1) * \sigma(\boldsymbol{r_o} \cdot \boldsymbol{r_s}), \text{otherwise} \end{cases} \tag{16}$$

where $W_1 \in \mathbb{R}^{1 \times d_m}$, $b_1 \in \mathbb{R}^1$, and d_m is the dimension of the semantic embedding. σ denotes sigmoid function.

3.3.5. Asymmetric Axiom Module

Asymmetric axiom module considers two aspects of judgements as well, which are the property of the relation *(i.e., whether r is asymmetric)* and the existence of symmetric triple of the given triple *(i.e., whether (s, r, o) and (o, r, s) both exist in the same KG)*. In OWL2, a relation can be asymmetric meaning that if it connects s with o it never connects o with s. In other words, if r is an asymmetric relation, and (s, r, o) and (o, r, s) appear in the same KG simultaneously, one of the two triples is incorrect for the violation of *asymmetric* axiom. For example, when the relation is *hasChild*, and there is a correct triple (*John, hasChild, David*), (*David, hasChild, John*) is incorrect because of the *asymmetric* axiom.

In this axiom, we firstly begin the process of determining the property of the relation by focusing on the semantic embedding $\boldsymbol{r_m}$. We take a multi-layer perceptron (MLP) layer to capture the semantics of the relation for further judgement of the property of the relation. Secondly, we use the simplest knowledge graph embedding model TransE to check whether (o, r, s) exists. Then, as only when the two conditions met at the same time, the violation of the axiom can be determined, otherwise the triple satisfies *asymmetric* axiom. We introduce a Logical AND operation as well. Finally, the *asymmetric* axiom network can be defined as follows:

$$f_{kge} = \sigma(\boldsymbol{o_m} + \boldsymbol{r_m} - \boldsymbol{s_m}) \tag{17}$$

$$P_{asy} = \sigma(W_2(\boldsymbol{r_m}) + b_2) * f_{kge} \tag{18}$$

where $W_2 \in \mathbb{R}^{1\times d_m}$, $b_2 \in \mathbb{R}^1$. σ denotes sigmoid function.

4. Experiments

We evaluate our proposed method through three main knowledge graph reasoning tasks, including noise detection, link prediction, and triple classification.

4.1. Experimental Settings

Datasets. In this paper, we use two popular benchmark datasets: FB15K237[37] and WN18RR[16]. They are constructed from FB15K and WN18 respectively by removing inverse relations to solve test leakage. FB15K is a relatively dense subset extracted from Freebase[2], which is a large collaborative knowledge graph consists of billions of real-world facts. And WN18 is a subset of WordNet[38] that describes relations between words.

Dataset	#Ent	#Rel	#Train	#Valid	#Test
FB15K237	14541	237	272115	17535	20466
WN18RR	40943	11	86845	3034	3134

Table 2

Statistics of FB15K237 and WN18RR

Datasets	FB15K237-10%	FB15K237-20%	FB15K237-40%
#Neg triple	27211	54423	108846
Datasets	WN18RR-10%	WN18RR-20%	WN18RR-40%
#Neg triple	8683	17367	34734

Table 3

Statistics of negative triples generated from FB15K237 and WN18RR

Error Imputation. Since in real world, KGs are constructed in an automated or semi-automated way, noises can not be avoided. However, there are no explicitly-labeled noisy triples in FB15K237 or WN18RR. Therefore, to assess the methodologies presented, we generate new datasets with different noise rates based on the two datasets to simulate the real noisy knowledge graphs. Specifically, for each dataset, we first randomly sample positive triples from the training set. The number of sampled triple is equal to the number of noisy triples that need to be generated. Then, for each positive triple (s, r, o), we corrupt either the subject entity or object entity with equal probability. If the former, we form a negative triple (s', r, o), otherwise the negative triple is (s, r, o'). The triples (s', r, o) and (s, r, o') are not in KGs. All the three tasks are evaluated on these simulated noisy datasets.

For each dataset, we construct three noisy datasets with negative triples to be 10%, 20%, and 40% of positive triples. And the negative triples will be imbued to the training set of the original dataset, and be regarded as positive triples for training. All the three noisy datasets share the same entities, relations, validation and test sets with the original dataset. The detailed statistics of the two datasets, and the generated noisy datasets are shown in Table 2 and 3.

Baselines. We choose TransE or TransH as the knowledge graph embedding module respectively, and compare our methods[*] NeuRAN(TransE) and NeuRAN(TransH) which introduce axiom modules with them. CKRL(TransE) and CKRL(TransH) are also considered as baselines, which focus on the utilization of path information. Results of TransE and TransH are produced by running OpenKE[39] with its default parameters. Results of CKRL(TransE) and CKRL(TransH) are reproduced by us. In the following

[*]The code of NeuRAN is available at https://github.com/JuanLi1621/NeuRAN

	FB15K237-10%	FB15K237-20%	FB15K237-40%	WN18RR-10%	WN18RR-20%	WN18RR-40%
TransE	0.9174	0.9431	0.9588	0.8403	0.8202	0.7961
CKRL(TransE)	0.9775	0.9755	0.9745	0.8584	0.8340	0.8015
NeuRAN(TransE)	**0.9811**	**0.9883**	**0.9796**	**0.9365**	**0.9233**	**0.8864**
TransH	0.8544	0.8986	0.9392	0.8191	0.7882	0.7748
CKRL(TransH)	**0.9787**	**0.9774**	**0.9763**	0.8412	0.8148	0.7805
NeuRAN(TransH)	0.9751	0.9740	0.9739	**0.8903**	**0.8751**	**0.8459**

Table 4

Noise detection results on noisy datasets with different ratios based on FB15K237 and WN18RR.

tasks, the results of TransE, TransH, CKRL(TransE) and CKRL(TransH) are also obtained in this way.

Training Details. We use SGD[40] or Adam[41] to optimize the model for different tasks on different datasets. We select the learning rate from {0.01, 0.1, 0.5, 1}, the margin among {2,4,6,8,10}, the batch size from {100, 500, 1000}, dimension of the type from {20, 50}, dimension of the semantic from {50, 100, 200}, the combination weight of the axiom scores α from {0.01, 0.05, 0.1, 0.5, 1}. The number of training epochs is set as 1000.

4.2. Noise Detection

To verify the capability of our method in detecting noises in a noisy knowledge graph, we use the evaluation task KG noise detection proposed in [30]. This task aims to detect possible noises in noisy KGs according to the scores of triples, which can be viewed as triple classification task on training set.

Evaluation Protocol. First of all, we compute the score of a triple via the energy function $E(s,r,o) = E_S + \alpha(E_{DM} + E_{RG} + E_{DIS} + E_{IRRE} + E_{ASYM})$. Then all triples in training set will be ranked based on the scores. The lower the score of the triple, the more valid the triple is. Triples with higher values of the energy function tend to be noises. We consider evaluation indicator the Area Under the ROC Curve (*auc value*) to examine how well the method classify the noise as an error. Before calculating the auc metric, we normalize the energy function score in the [0, 1] interval, values close to 0 indicate a correct triple, and values close to 1 indicate an erroneous triple.

Result Analysis. Evaluation results on noisy datasets generated based on FB15K237 and WN18RR can be found in Table 4. We observe that: (1) Regardless of the embedding module is TransE or TransH, our model achieves the best performance on WN18RR with different noise rates (i.e., WN18RR-10%, WN18RR-20% and WN18RR-40%), demonstrating the capability of our model in detecting noises. (2) On FB15K237-10%, FB15K237-20% and FB15K237-40%, results of our method outperform TransE, CKRL(TransE) and TransH, but are slightly worse than CKRL(TransH). It indicates both path information and axiom information can help noise detection. And when the complex relations are well encoded, the larger the number of relations and triples, the path information may be more effective than axiom information on noise detection. (3) On WN18RR-10%, WN18RR-20% and WN18RR-40% datasets, our results are significantly improved compared with CKRL(TransE) and CKRL(TransH). But on FB15K237-based noisy datasets, the results are not competitive. It may not only because FB15K237-based noisy datasets contain much more relations and triples than WN18RR-based, but also due to that relations in WN18RR-based dataset are difficult to form path information. While as each relation has domain and range axioms, it is reasonable that axiom information can have good noise detection capabilities when there are only a few relations in a dataset. (4) With the increase of noises, the ability of baselines and our model to detect noises decreases on WN18RR-based datasets. But may increase on FB15K237-based datasets, it indicates when the number of relations is large, the more triples in noisy datasets, the more valid information may be introduced, even these triples may be noises.

Thus we conclude that implicit axiom information is useful for noise detection, and the improvements are more obvious especially in small datasets with a small number of relations and triples.

4.3. Triple Classification

Triple classification aims to judge whether a triple in test data is correct or not according to triple scores calculated by the energy function $E(s,r,o) = E_S + \alpha(E_{DM} + E_{RG} + E_{DIS} + E_{IRRE} + E_{ASYM})$, which can be viewed as a binary classification task on the test set.

Methods	FB15K237-10%			FB15K237-20%			FB15K237-40%		
	ACC	P	R	ACC	P	R	ACC	P	R
TransE	93.91	92.51	95.55	93.12	93.02	93.23	92.61	91.95	93.39
CKRL(TransE)	94.43	92.61	**96.56**	94.15	92.37	96.23	93.39	91.56	95.59
NeuRAN(TransE)	**94.92**	**93.55**	96.50	**94.77**	**93.04**	**96.79**	**94.33**	**92.78**	**96.15**
TransH	93.59	93.73	93.43	93.28	93.03	93.58	92.81	92.65	93.00
CKRL(TransH)	94.78	**94.05**	95.60	94.51	93.46	95.72	94.03	**93.17**	95.04
NeuRAN(TransH)	**95.16**	93.77	**96.75**	**94.94**	**93.63**	**96.44**	**94.52**	93.10	**96.16**

Table 5

triple classification results for FB15K237-10%, FB15K237-20% and FB15K237-40%.

Methods	WN18RR-10%			WN18RR-20%			WN18RR-40%		
	ACC	P	R	ACC	P	R	ACC	P	R
TransE	87.73	92.46	82.16	86.38	89.18	82.80	83.49	88.34	77.15
CKRL(TransE)	88.19	93.31	82.29	86.49	90.71	81.30	83.31	85.80	79.83
NeuRAN(TransE)	**89.29**	**94.83**	**83.12**	**87.84**	**91.35**	**83.60**	**85.90**	**90.24**	**80.50**
TransH	86.87	90.64	**82.23**	84.99	86.03	**83.54**	82.29	83.48	**80.50**
CKRL(TransH)	86.65	92.40	79.87	85.63	90.70	79.39	82.35	83.64	80.44
NeuRAN(TransH)	**87.92**	**93.61**	81.40	**86.95**	**91.90**	81.05	**84.43**	**87.52**	80.31

Table 6

triple classification results on WN18RR, WN18RR-10%, WN18RR-20% and WN18RR-40%.

Evaluation Protocol. As the test set of the datasets used for triple classification only contains correct triples, we construct negative triples by corrupt the subject or object entity of correct triples. There are as many positive triples as negative triples in both valid and test set. For triple classification, we learn a relation-specific threshold δ_r for every relation. δ_r is optimized by maximizing classification accuracies on the validation set. Given a triple (s, r, o), if the score obtained by the energy function is below δ_r, it is classified as positive, otherwise negative. We use accuracy(ACC), precision(P) and recall(R) as the evaluation metrics.

Result Analysis. Table 5 and 6 show the detailed evaluation results of triple classification. From the two tables, we can observe that: (1) In terms of the three metric, our method outperforms baselines on the WN18RR-based and FB15K237-based datasets except for few cases, and it achieves the best results in accuracy metric. This confirms that learning knowledge representations with axiom information could help for triple classification as well. (2) Whether on WN18RR-10%, WN18RR-20% and WN18RR-40%, or on FB15K237-10%, FB15K237-20% and FB15K237-40%, the advantages our method have over baselines in this task seem to be smaller than those in noise detection. It may because the method concentrates on negative triples in training set, but not on negative triples that generated in test set. (3) Compared with TransE, TransH, CKRL(TransE) and CKRL(TransH), the improvements of results in accuracy metric on FB15K237-10%, FB15K237-20% and FB15K237-40%, as well as on WN18RR-10%, WN18RR-20% and WN18RR-40% are more larger with higher noise rates. It indicates that on noisy datasets, triple classification results of NeuRAN can be more robust than baselines regardless of the size of the dataset.

From the results, we can conclude that in triple classification task, combining implicit axiom and structural information reflected by existing triples in knowledge graphs works better than using structural information only. And the combination helps to learn robust embeddings on noisy datasets with a large or small number of relations and triples.

4.4. Link Prediction

To show that axiom information could improve embedding learning of entities and relations, and further help complete knowledge graphs, we conduct link prediction task to evaluate the performance of knowledge

	FB15K237-10%			FB15K237-20%			FB15K237-40%		
	MRR	Hit@		MRR	Hit@		MRR	Hit@	
		3	1		3	1		3	1
TransE	0.259	0.298	0.161	0.242	0.281	0.146	0.228	0.267	0.134
CKRL(TransE)	0.261	0.292	0.175	0.256	0.286	0.172	**0.250**	**0.280**	0.167
NeuRAN(TransE)	**0.285**	**0.312**	**0.200**	**0.269**	**0.293**	**0.187**	**0.250**	0.271	**0.174**
TransH	0.244	0.297	0.129	0.214	0.269	0.096	0.191	0.247	0.075
CKRL(TransH)	0.269	0.303	0.176	0.265	0.294	0.181	0.258	**0.288**	0.175
NeuRAN(TransH)	**0.287**	**0.316**	**0.203**	**0.273**	**0.297**	**0.190**	**0.259**	0.282	**0.181**

Table 7

Link prediction results on FB15K237-10%, FB15K237-20% and FB15K237-40%.

	WN18RR-10%			WN18RR-20%			WN18RR-40%		
	MRR	Hit@		MRR	Hit@		MRR	Hit@	
		3	1		3	1		3	1
TransE	0.212	0.348	0.036	0.204	0.349	0.029	0.190	0.324	0.027
CKRL(TransE)	0.216	0.348	0.043	0.210	0.356	0.036	0.187	0.321	0.026
NeuRAN(TransE)	**0.339**	**0.387**	**0.269**	**0.319**	**0.369**	**0.245**	**0.268**	**0.339**	**0.173**
TransH	0.218	0.362	0.042	0.208	0.357	0.035	0.193	0.331	0.032
CKRL(TransH)	0.221	0.367	0.043	0.209	**0.358**	0.037	0.194	0.330	0.036
NeuRAN(TransH)	**0.312**	**0.379**	**0.223**	**0.262**	0.354	**0.147**	**0.295**	**0.348**	**0.214**

Table 8

Link prediction results on WN18RR-10%, WN18RR-20% and WN18RR-40%.

graph completion. This task aims to predict the missing entity when given one entity and the relation in a triple, including subject entity prediction $(?, r, o)$ and object entity prediction $(s, r, ?)$.

Evaluation Protocol. For each test triple, suppose the subject entity prediction (?,r,o) with the right subject entity s. We first take all entities $e \in \mathcal{E}$ in the dataset as candidate predictions, and then replace the missing part with each entity e and calculate scores for triples in $\mathcal{T} = \{(e, r, o)|e \in \mathcal{G}\}$. Subsequently, we rank these scores by ascending order, the rank of the correct entity is stored. The object entity prediction is done in the same way. The evaluation metrics are MRR and Hits@N, where MRR is the mean reciprocal rank of the ranks of all test triples, and Hits@N (N=1,3) is the proportion of ranks within N of all test triples. A higher MRR and a higher Hits@1, 3 should be achieved by a good embedding model. This is called 'raw' setting. If we filter out the corrupted triples that exist in the training, validation, or test set before ranking, the evaluation setting is called 'filter'. In this paper, we report evaluation results of the filter setting.

Result Analysis. Link prediction results are shown in Table 7 and 8. We analyze the results as follows: (1) The link prediction results of our method are improved compared with baselines on WN18RR-10%, WN18RR-20% and WN18RR-40% datasets, as well as on FB15K237-10%, FB15K237-20% and FB15K237-40%. It confirms that the quality of learned knowledge graph embeddings are better, and could also help to complete KGs. Besides, it indicates axiom information can be more useful than path information on noisy datasets. (2) On WN18RR-10%, WN18RR-20% and WN18RR-40% datasets, our method achieves the best performance almost on all metrics. And the improvements are significantly, especially on Hit@1. It demonstrates that axiom information is of great help in improving the predictive ability a missing triple, when a dataset has fewer relations and triples. (3) On FB15K237-10%, FB15K237-20% and FB15K237-40%, although the improvements of the results are less obvious compared with WN18RR-based datasets, the results are better than baselines. Moreover, as the noise ratio increases, most results of our method decreases more slowly compared to baselines. It reaffirms that our method can improve link prediction, and the more relations and triples, the more information as well as noises brought by axiom information. Therefore, the advantages of implicit axiom information would not as significant as in small-scale dataset.

Thus we can conclude that implicit axiom information encoded by neural axiom networks help improve the quality of learned embeddings of entities and relations, and improve link prediction. And such information is more effective on datasets with a relatively small number of relations and triples.

5. Conclusion

In this paper, we propose a novel neural axiom network model which aims to do reasoning on noisy knowledge graphs. We consider to encode not only structural information, but also axiom information of triples. In specific, we propose a knowledge graph embedding module for preserving the structure, and five different axiom modules for calculating probability scores that the corresponding axioms are satisfied. We evaluate our method on KG noise detection, triple classification and link prediction. Experiments show that axiom information can benefit these tasks.

In the future, we will attempt to explore more implicit or explicit information existed in triples which can enhance the performance of knowledge graph reasoning. Further more, we will improve our method to apply it for inconsistency reasoning, due to that axiom information may be able to provide explanations for inconsistent triples, so as to correct erroneous triples.

References

[1] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P.N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer and C. Bizer, DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia, *Semantic Web* **6**(2) (2015), 167–195.

[2] K.D. Bollacker, C. Evans, P. Paritosh, T. Sturge and J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: *SIGMOD Conference*, ACM, 2008, pp. 1247–1250.

[3] T.P. Tanon, D. Vrandecic, S. Schaffert, T. Steiner and L. Pintscher, From Freebase to Wikidata: The Great Migration, in: *WWW*, ACM, 2016, pp. 1419–1428.

[4] J. Hoffart, F.M. Suchanek, K. Berberich and G. Weikum, YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia, *Artif. Intell.* **194** (2013), 28–61.

[5] A. Bordes, J. Weston and N. Usunier, Open Question Answering with Weakly Supervised Embedding Models, in: *ECML/PKDD (1)*, Lecture Notes in Computer Science, Vol. 8724, Springer, 2014, pp. 165–180.

[6] X. Wang, X. He, Y. Cao, M. Liu and T. Chua, KGAT: Knowledge Graph Attention Network for Recommendation, in: *KDD*, ACM, 2019, pp. 950–958.

[7] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta and D. Lin, Knowledge base completion via search-based question answering, in: *WWW*, ACM, 2014, pp. 515–526.

[8] A. Melo and H. Paulheim, Automatic detection of relation assertion errors and induction of relation constraints, *Semantic Web* **11**(5) (2020), 801–830.

[9] L.A. Galárraga, C. Teflioudi, K. Hose and 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.

[10] L. Galárraga, C. Teflioudi, K. Hose and F.M. Suchanek, Fast rule mining in ontological knowledge bases with AMIE+, *VLDB J.* **24**(6) (2015), 707–730.

[11] W.W. Cohen, TensorLog: A Differentiable Deductive Database, *CoRR* **abs/1605.06523** (2016).

[12] F. Yang, Z. Yang and W.W. Cohen, Differentiable Learning of Logical Rules for Knowledge Base Reasoning, in: *NIPS*, 2017, pp. 2319–2328.

[13] A. Bordes, N. Usunier, A. García-Durán, J. Weston and O. Yakhnenko, Translating Embeddings for Modeling Multi-relational Data, in: *NIPS*, 2013, pp. 2787–2795.

[14] B. Yang, W. Yih, X. He, J. Gao and L. Deng, Embedding Entities and Relations for Learning and Inference in Knowledge Bases, in: *ICLR (Poster)*, 2015.

[15] M. Nickel, L. Rosasco and T.A. Poggio, Holographic Embeddings of Knowledge Graphs, in: *AAAI*, AAAI Press, 2016, pp. 1955–1961.

[16] T. Dettmers, P. Minervini, P. Stenetorp and S. Riedel, Convolutional 2D Knowledge Graph Embeddings, in: *AAAI*, AAAI Press, 2018, pp. 1811–1818.

[17] Z. Wang, J. Zhang, J. Feng and Z. Chen, Knowledge Graph Embedding by Translating on Hyperplanes, in: *AAAI*, AAAI Press, 2014, pp. 1112–1119.

[18] Y. Lin, Z. Liu, M. Sun, Y. Liu and X. Zhu, Learning Entity and Relation Embeddings for Knowledge Graph Completion, in: *AAAI*, AAAI Press, 2015, pp. 2181–2187.

[19] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier and G. Bouchard, Complex Embeddings for Simple Link Prediction, in: *ICML*, JMLR Workshop and Conference Proceedings, Vol. 48, JMLR.org, 2016, pp. 2071–2080.

[20] D.Q. Nguyen, T.D. Nguyen, D.Q. Nguyen and D.Q. Phung, A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network, in: *NAACL-HLT (2)*, Association for Computational Linguistics, 2018, pp. 327–333.

[21] D.Q. Nguyen, T. Vu, T.D. Nguyen, D.Q. Nguyen and D.Q. Phung, A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization, in: *NAACL-HLT (1)*, Association for Computational Linguistics, 2019, pp. 2180–2189.

[22] R. Irny and P.S. Kumar, Mining Inverse and Symmetric Axioms in Linked Data, in: *JIST*, Lecture Notes in Computer Science, Vol. 10675, Springer, 2017, pp. 215–231.

[23] D. Fleischhacker and J. Völker, Inductive Learning of Disjointness Axioms, in: *OTM Conferences (2)*, Lecture Notes in Computer Science, Vol. 7045, Springer, 2011, pp. 680–697.

[24] Y. Ma, H. Gao, T. Wu and G. Qi, Learning Disjointness Axioms With Association Rule Mining and Its Application to Inconsistency Detection of Linked Data, in: *CSWS*, Communications in Computer and Information Science, Vol. 480, Springer, 2014, pp. 29–41.

[25] G. Töpper, M. Knuth and H. Sack, DBpedia ontology enrichment for inconsistency detection, in: *I-SEMANTICS*, ACM, 2012, pp. 33–40.

[26] G. Ji, S. He, L. Xu, K. Liu and J. Zhao, Knowledge Graph Embedding via Dynamic Mapping Matrix, in: *ACL (1)*, The Association for Computer Linguistics, 2015, pp. 687–696.

[27] T. Ebisu and R. Ichise, TorusE: Knowledge Graph Embedding on a Lie Group, in: *AAAI*, AAAI Press, 2018, pp. 1819–1826.

[28] Z. Sun, Z. Deng, J. Nie and J. Tang, RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space, in: *ICLR (Poster)*, OpenReview.net, 2019.

[29] S. Jia, Y. Xiang, X. Chen, K. Wang and S. E, Triple Trustworthiness Measurement for Knowledge Graph, in: *WWW*, ACM, 2019, pp. 2865–2871.

[30] R. Xie, Z. Liu, F. Lin and L. Lin, Does William Shakespeare REALLY Write Hamlet? Knowledge Representation Learning With Confidence, in: *AAAI*, AAAI Press, 2018, pp. 4954–4961.

[31] X. Lv, L. Hou, J. Li and Z. Liu, Differentiating Concepts and Instances for Knowledge Graph Embedding, in: *EMNLP*, Association for Computational Linguistics, 2018, pp. 1971–1979.

[32] L. Zhao, X. Zhang, K. Wang, Z. Feng and Z. Wang, Learning Ontology Axioms over Knowledge Graphs via Representation Learning, in: *ISWC Satellites*, CEUR Workshop Proceedings, Vol. 2456, CEUR-WS.org, 2019, pp. 57–60.

[33] W. Zhang, B. Paudel, L. Wang, J. Chen, H. Zhu, W. Zhang, A. Bernstein and H. Chen, Iteratively Learning Embeddings and Rules for Knowledge Graph Reasoning, in: *WWW*, ACM, 2019, pp. 2366–2377.

[34] H. Paulheim and C. Bizer, Improving the Quality of Linked Data Using Statistical Distributions, *Int. J. Semantic Web Inf. Syst.* **10**(2) (2014), 63–86.

[35] Q. Wang, Z. Mao, B. Wang and L. Guo, Knowledge Graph Embedding: A Survey of Approaches and Applications, *IEEE Trans. Knowl. Data Eng.* **29**(12) (2017), 2724–2743.

[36] P. Jain, P. Kumar, Mausam and S. Chakrabarti, Type-Sensitive Knowledge Base Inference Without Explicit Type Supervision, in: *ACL (2)*, Association for Computational Linguistics, 2018, pp. 75–80.

[37] K. Toutanova and D. Chen, Observed versus latent features for knowledge base and text inference, in: *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, 2015, pp. 57–66.

[38] G.A. Miller, WordNet: A Lexical Database for English, *Commun. ACM* **38**(11) (1995), 39–41.

[39] X. Han, S. Cao, X. Lv, Y. Lin, Z. Liu, M. Sun and J. Li, OpenKE: An Open Toolkit for Knowledge Embedding, in: *EMNLP (Demonstration)*, Association for Computational Linguistics, 2018, pp. 139–144.

[40] J.C. Duchi, E. Hazan and Y. Singer, Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, *J. Mach. Learn. Res.* **12** (2011), 2121–2159.

[41] D.P. Kingma and J. Ba, Adam: A Method for Stochastic Optimization, in: *ICLR (Poster)*, 2015.