Neural Axiom Network for Knowledge Graph Reasoning

Juan Li\textsuperscript{a}, Xiangnan Chen\textsuperscript{a}, Hongtao Yu\textsuperscript{a}, Jiaoyan Chen\textsuperscript{b} and Wen Zhang\textsuperscript{c}\textsuperscript{*}

\textsuperscript{a} College of Computer Science and Technology, Zhejiang University, 38 Zheda Rd, Hangzhou, China
\textit{E-mails: lijuan18@zju.edu.cn, xnchen2020@zju.edu.cn, yuhongtaoaaa@zju.edu.cn}
\textsuperscript{b} Department of Computer Science, University of Oxford, 15 Parks Rd, Oxford OX1 3QD, UK
\textit{E-mail: jiaoyan.chen@cs.ox.ac.uk}
\textsuperscript{c} School of Software Technology, Zhejiang University, 38 Zheda Rd, Hangzhou, China
\textit{E-mail: zhang.wen@zju.edu.cn}

Abstract. Knowledge graphs (KGs) generally suffer from incompleteness and incorrectness problems due to the automatic and semi-automatic construction process. Knowledge graph reasoning aims to infer new knowledge or detect noises, which is essential for improving the quality of knowledge graphs. In recent years, various KG reasoning techniques, such as symbolic- and embedding-based methods, have been proposed and shown strong reasoning ability. Symbolic-based reasoning methods infer missing triples according to predefined rules or ontologies. Although rules and axioms have proven to be effective, it is difficult to obtain them. While embedding-based reasoning methods represent entities and relations of a KG as vectors, and complete the KG via vector computation. However, they mainly rely on structural information, and ignore implicit axiom information that are not predefined in KGs but can be reflected from data. That is, each correct triple is also a logically consistent triple, and satisfies all axioms. In this paper, we propose a novel \textbf{NeuRal Axiom Network (NeuRAN)} framework that combines explicit structural and implicit axiom information. It only uses existing triples in KGs without introducing additional ontologies. Specifically, the framework consists of a knowledge graph embedding module that preserves the semantics of triples, and five axiom modules that encode five kinds of implicit axioms using entities and relations in triples. These axioms correspond to five typical object property expression axioms defined in OWL2, including \texttt{ObjectPropertyDomain}, \texttt{ObjectPropertyRange}, \texttt{DisjointObjectProperties}, \texttt{IrreflexiveObjectProperty} and \texttt{AsymmetricObjectProperty}. The knowledge graph embedding module and axiom modules respectively compute the scores that the triple conforms to the semantics and the corresponding axioms. Evaluations on KG reasoning tasks show the efficiency of our method. Compared with knowledge graph embedding models and CRKL, our method achieves comparable performance on noise detection and triple classification, and achieves significant performance on link prediction. Compared with TransE and TransH, our method improves the link prediction performance on the Hit@1 metric by 22.4\% and 21.2\% on WN18RR-10\% dataset respectively.

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 \((\text{subject entity}, \text{relation}, \text{object entity})\), 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 constructed 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,
Fig. 1. In the hypothetical knowledge graph, there may exist erroneous triples. 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.

many knowledge graph reasoning methods have been proposed and received increasing attention. There are two mainstream techniques, including symbolic- and embedding-based methods.

Symbolic-based methods [9–12] use given logic rules or ontologies for KG reasoning, and can achieve good performance. For example, suppose an axiom DisjointObjectProperties(:hasParent :hasSpouse) has been already defined in a KG, indicating that the relations hasSpouse and hasParent are disjoint. For the two triples (Linda, hasSpouse, Bruce) and (Linda, hasParent, Bruce) in the KG, if the former is correct, the latter will be classified as incorrect. The reason is that a person’s spouse can not be the parent of this person. Although such methods are more reliable and human-interpretable, they require rich ontologies which are usually missing or incomplete in KGs. Moreover, it is tedious to define and maintain axioms manually. Thus, we explore how to encode implicit axioms with only triples for KG reasoning in this paper.

Embedding-based methods, such as translation-based methods [13–15], semantic-based methods [16, 17] and neural network methods [18–20], embed entities and relations into low-dimensional vector space. They use vector computation to complete knowledge graphs, which is scalable and efficient. However, despite the success of embedding models, most of them focus on structural information without taking implicit axiom information into consideration. Take implicit domain/range axiom as an example. For the correct triple (Linda, hasSpouse, Bruce), we can infer that it satisfies domain/range axiom without type of the subject entity Linda, type of the object entity Bruce, and domain/range of the relation hasSpouse.

In this paper, we propose a neural axiom network framework NeuRAN for KG reasoning. This framework encodes not only 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. Thus, the score to measure the plausibility of each triple is composed of a score from structural information and five axiom scores from axiom information. Here we consider five different axioms corresponding to five typical object property expression axioms selected from OWL2 ontology language*.

* https://www.w3.org/TR/owl2-primer/
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. \(\Delta\) is a nonempty set called the object domain. \(\Delta^*\) is an object property interpretation function. When translating axioms into examples in KG according to condition, we replace OPE in axioms with a relation.

<table>
<thead>
<tr>
<th>Object Property Axioms</th>
<th>Condition</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPDomain(OPE CE)</td>
<td>(\forall(x, y) \in (OPE)^* \implies x \in (CE)^*)</td>
<td>Domain(hasWife, Man)</td>
</tr>
<tr>
<td>OPRange(OPE CE)</td>
<td>(\forall(x, y) \in (OPE)^* \implies y \in (CE)^*)</td>
<td>Range(hasWife, Woman)</td>
</tr>
<tr>
<td>Disjoint(OPE1, OPE2)</td>
<td>((OPE1)^* \cap (OPE2)^* = \emptyset) for each (1 \leq j \leq n) and each (1 \leq k \leq n) such that (j \neq k)</td>
<td>Disjoint(hasParent, hasSpouse)</td>
</tr>
<tr>
<td>Irreflexive(OPE)</td>
<td>(\forall x \in \Delta), (\forall z \in \Delta^<em>), (\forall (x, z) \notin (OPE)^</em>)</td>
<td>Irreflexive(hasSibling)</td>
</tr>
<tr>
<td>Asymmetric(OPE)</td>
<td>(\forall (x, y) \in (OPE)^* \implies (y, x) \notin (OPE)^*)</td>
<td>Asymmetric(hasParent)</td>
</tr>
</tbody>
</table>

2. Related work

We discuss the following three lines of research work that are closely relevant to this paper, including symbolic-based reasoning, embedding-based reasoning and hybrid reasoning.

2.1. Symbolic-based Reasoning

Symbolic-based reasoning methods aim at inferring new knowledge or detecting noises with the help of rules or ontologies, and show good reasoning ability. Due to the incompleteness of rules and axioms, and the time-consuming process of annotating them, existing methods of using ontologies for reasoning are usually accompanied by enrichment of ontology information. For example, inductive logic programing (ILP) has been used to mine logical rules. But it has limitations on the 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+, [21] proposes to discover inverse and symmetric axioms by applying the predefined reasoning rules. Moreover, to enrich ontologies with disjointness axioms, [22] presents a set of inductive methods based on statistical inductive learning, including correlation computing and association rule mining. As it evaluates the validity of association rule mining by...
computing the precision and recall scores, [23] not only discusses the precision and recall, but also analyzes quality of disjoint axioms acquired. In addition to disjoint axioms, enriching DBpedia ontology with domain and range restrictions, as well as class disjointness axioms is also discussed [24]. The enhanced ontologies are further used for error detection.

2.2. Embedding-based Reasoning

Knowledge graph embedding methods embed 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 \((s + r)\) to be close to \(o\), where \(s, r, o\) are vector embeddings for \(s, r\) and \(o\) respectively. However, it has difficulty dealing with complex relations. To overcome the flaws, TransH [14] introduces relation-specific hyperplanes to allow entities have different embeddings in different relations. TransR [15] builds entity and relation embeddings in separate entity and relation spaces. And TransD [25] constructs mapping matrices dynamically. Similarly, TorusE [26] and RotatE [27] use lie groups and rotations for translation respectively. Semantic matching models measure plausibility by matching implicit semantics of entities and relations. DistMult [16] uses a formulation of bilinear model to represent entities and relations. ComplEx [17] extends DistMult by introducing complex-valued embeddings so as to better model asymmetric relations. HolE [28] makes use of circular correlation of embeddings to learn compositional representations and semantically matches circular correlation with the relation embedding. Moreover, researchers have raised interests in applying neural networks for knowledge graph reasoning. ConvE [18] and ConvKB [19] employ convolutional neural network to achieve better link prediction performance. CapsE [20] explores a capsule network to model triples. Moreover, 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- and embedding-based reasoning, and the combination of symbolic and statistical reasoning. For the former methods, TransC [31] learns SubClassOf axiom between types by encoding each type as a sphere and each entity as a vector. Besides, 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 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. Furthermore, [35] first exploits a combination of entailment vectors, entailment weights, and a consistency vector to encode knowledge as embeddings in ontology streams to deal with concept drifts. Concept is taken as input to its prediction model. The difference is that axioms are not given in our method. And we design the axiom modules with the help of the definition of axioms.

3. Method

We begin this section by briefly describing some notations. We denote a knowledge graph as \(G = \{(s, r, o) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}\), 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 indicating the relation \(r\) between the subject entity \(s\) and the object entity \(o\). Throughout this paper, we use bold letters to denote vectors. For example, \(s, r, o\) are the embedding vectors of \(s, r, o\). The abbreviations \(DM(dm)\), \(RG(rg)\), \(DIS(dis)\), \(IRRE(irre)\) and \(ASY(asy)\) 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). Afterwards, 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

We present our neural axiom network NeuRAN in Figure 2, where the overall score of each triple is composed of a semantic score from KGE module, and five axiom scores from five axiom modules (i.e., domain, range, disjoint, irreflexive and asymmetric axiom modules). For each axiom module, the score indicates the probability that the axiom holds. The assumption is that probability values of these axioms intensify or mitigate the probability of existence of a triple. Thus, the score of the triple \((s, r, o)\) is defined as:

\[
E(s, r, o) = E_{kge} + \lambda \cdot [(1 - P_{dm}) + (1 - P_{rg})]
+ (1 - P_{dis}) + (1 - P_{irr}) + (1 - P_{asy})]
\]

The energy function \(E(s, r, o)\) consists the score \(E_{kge}\) from the knowledge graph module and five axiom scores from the axiom modules. \(\lambda\) is the weight of axiom scores. \(E_{kge}\) is to compute a structure-based score, which can be obtained via any knowledge graph embedding models [36]. TransE and TransH are considered as examples. A lower \(E_{kge}\) indicates that the triple is more likely to be correct. The axiom scores \(P_{dm}, P_{rg}, P_{dis}, P_{irr}\) and \(P_{asy}\) correspond to probabilities that the corresponding axioms are satisfied, where \(dm, rg, dis, irr\) and \(asy\) are respectively domain, range, disjoint, irreflexive and asymmetric axioms. Therefore, the higher \(P_{dm}, P_{rg}, P_{dis}, P_{irr}, P_{asy}\) and the lower \((1 - P_{dm}), (1 - P_{rg}), (1 - P_{dis}), (1 - P_{irr}), (1 - P_{asy})\) imply that the triple satisfies the corresponding neural axiom with a higher probability.
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 regarded as positive triples, while the unobserved ones are regarded as negative triples. We utilize a margin-based ranking loss on pair-wise score functions (i.e., $E(s, r, o)$ and $E(s', r', o')$) for training, the loss function $L$ is defined as:

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

where $\gamma$ is a margin hyper-parameter, $E(s, r, o)$ and $E(s', r', o')$ are respectively the overall energy function of the positive triple $(s, r, o)$ and the negative triple $(s', r', o')$. $T$ and $T'$ are the positive and negative triple sets. As negative triples and axioms are not given, we generate negative triples by randomly corrupting the subject or object entity, and make sure the replaced triples do not exist in the knowledge graph:

$$
T' = \{ (s', r, o) | s' \in E \} \cup \{ (s, r', o') | o' \in E \},
$$

$$(s, r, o) \in T, (s', r, o) \notin T \land (s, r', o') \notin T \tag{3}
$$

The training objective is to minimize the loss function $L$ to learn embeddings of entities and relations, as well as parameters involved in axiom modules. The learned embeddings and parameters are applied to complete downstream tasks, such as noise detection, triple classification and link prediction.

It is worth mentioning that, we attempt to introduce type embeddings in the design of the model. Considering domain/range axioms are associated with type compatibility, and type information is not provided, we distinguish type embeddings from semantic embeddings. Following the type-sensitive models TypeDM and TypeComplex [37], each entity is represented as two vectors (one type embedding and one semantic embedding), and each relation is represented as three vectors (two type embeddings and one semantic embedding). Take the triple $(s, r, o)$ as an example. For entities, we use type embeddings $(s_e$ and $o_e$) and semantic embeddings $(s_m$ and $o_m$) to represent the subject entity $s$ and the object entity $o$. For the relation, $r_s$ and $r_o$ represent subject type and object type embeddings expected by the relation $r$, and $r_m$ 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 $E_{kg}$, which is designed to score each triple based on the structural information in KGS. Our framework can use any knowledge graph embedding model as the knowledge graph embedding module. In the experiments, we take the two embedding models TransE and TransH as examples to verify our method.

#### 3.2.1. TransE

TransE is the simplest translation-based model, which interprets relations as translating operations between subject and object entities. Given a triple $(s, r, o)$, it follows the assumption that $s + r \approx o$ when $(s, r, o)$ holds. Thus, the semantic score of the triple based on TransE is calculated as:

$$
E_{kg} = ||s_m + r_m - o_m||_{L_1/L_2} \tag{4}
$$

where $L_1$ and $L_2$ respectively denote the $L_1$ and $L_2$ norm. $s_m$, $r_m$ and $o_m$ are the semantic embeddings of the subject entity, the relation and the object entity respectively. The smaller value of the scoring function $E_{kg}$, the higher the probability that the triple is correct.

#### 3.2.2. TransH

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

$$
E_{kg} = ||(s_m) \perp + r_m - (o_m) \perp||_{L_1/L_2} \tag{5}
$$

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

### 3.3. Five Axiom Modules

In addition to the focus on structural information, we also consider the inherent implicit axioms that exist in triples. Although axioms are not pre-given, it is intuitive that a correct triple is also a logically consistent triple. That is any correct triple satisfies all the five axioms, including domain, range, disjoint, irreflexive, and asymmetric axioms. For example, in the case of missing domain and range of the relation nationalty, and types of the entities J. K. Rowling and United...
Kingdom, we can infer the triple (J. K. Rowling, nationality, United Kingdom) satisfies domain and range axioms by its correctness.

The design of the axiom modules is based on the definition of these axioms. To be specific, for domain/range axioms, the type embedding of the subject/object entity expected by the relation \( r_1 / r_a \) and the type embedding of the subject/object entity \( s_s / o_c \) 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 \((o_m - s_m)\) represents the shared semantic embedding of the relation for the subject to be \( s \) and object to be \( o \). Then the similar score of \( r_m \) and \((o_m - s_m)\) will be calculated to reflect how well the disjoint axiom is satisfied. 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 \((o, r, s) \in G^5\) are concerned. We then 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 the type of the subject entity \( s \). TypeDM uses the function \( C(s_e, r_s) = \sigma(s_e \cdot r_s) \) to measure the compatibility by calculating a score with the type embedding of subject entity \( s_e \) and the subject entity type embedding expected by the relation \( r_s \).

However, the subject type expected by the given relation may be diverse. For example, given the triple \( \text{Soul(film), language, English} \), the subject type expected by the relation language can be an entity in the class of Person, Book or Film. To capture type of the subject entity expected by the relation precisely, we consider both the current relation and the other relations of the subject entity. Suppose the subject entity has relations starring and running time, we can infer that the subject entity expected by the relation is more likely to be in the class of Film. As the relations may have strong correlations with the given relation, such as starring and language, or differ greatly such as running time and language, they contribute differently to the subject entity type embedding expected by the current given relation. We apply an attention mechanism to the relation \( r \) and the relation set of the subject entity \( \hat{R}(s) = \{ r_i | (s, r_i, e) \in G \} \), where \( e \) denotes any entity in the KG. By adopting a domain attention layer \( \text{Dm}_\text{Att}_\text{Layer} \), we generate the subject type embedding expected by the relation \( \hat{r}_s \) based on the relations in \( \hat{R}(s) \). We compute an attention weight for each relation of the subject entity, and the importance is denoted by \( a_i \). It reflects how relevant or important the relation \( r_i \) is to \( r_s \).

\[
a_i = f(r_s, r_i) = r^T \sigma r_i, r_i \in \hat{R}(s)
\]

(6)

To get the relative attention values, softmax is applied over \( a_i \).

\[
p_i = \frac{exp(a_i)}{\sum_{j \in \hat{R}(s)} exp(a_j)}
\]

(7)

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

\[
\hat{r}_s = \sum_{r_i \in \hat{R}(s)} p_i r_i
\]

(8)

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

\[
P_{dm} = f(s_e, r_s) = \sigma(s_e \cdot \hat{r}_s)
\]

(9)

where \( \sigma \) 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 \( C(o_c, r_o) = \sigma(o_c \cdot r_o) \) to compute the compatibility score between the type embedding of the object entity \( o_c \) and the object entity type embedding expected by the relation \( r_o \), where the score indicates 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 (Atlantic, hasPart, NewYorkBay). Similar to the issue in domain axiom, object entities expected by a relation may exhibit diverse roles within the same relation. The other relations connected to the object entity make different contributions to the embedding of the object entity type expected by the relation. We apply a range attention layer \( \text{Rg}_\text{Att}_\text{Layer} \) to the relations that connected to the object entity to discern the expected ob-
ject entity type associated with the given relation more
accurately. The relation set that connected to the object
entity o is denoted as $R(o) = \{r_i | (e, r_i, o) \in G\}$. We
generate the object entity type embedding expected by
the relation $\hat{r}_o$ based on all the relations in $R(o)$. The
importance of each relation to $r$ denoted by $b_i$ can be
calculated as:

$$b_i = f(r_o, r_i) = r_o^T r_i, r_i \in R(o)$$

(10)

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

$$q_i = \frac{\exp(b_i)}{\sum_{r_i \in R(o)} \exp(b_i)}$$

(11)

where $k$ denotes the $k$th relation in the connected relations of the object entity. The generated embedding $\hat{r}_o$ is the sum of the product of each relation connected to $o$ and the relation weighted by attention values of the considered relation $r$.

$$\hat{r}_o = \sum_{r_i \in R(o)} q_i r_i$$

(12)

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

$$P_{rg} = f(o_c, r_o) = \sigma(o_c \cdot \hat{r}_o)$$

(13)

where $\sigma$ 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, for the target correct triple (John, spouse, Mary), and the other two triples (John, friend, Mary) and (John, child, Mary), the disjoint axiom module computes compatibility scores of the two relation pairs including (spouse, friend), and (spouse, child). Then we can infer (John, friend, Mary) is a correct triple, and (John, child, Mary) is an incorrect triple. The reason is that the disjoint probability score of spouse and friend is high as they can exist between two persons at the same time. While the score of spouse and child is low, since a person’s spouse can not be that person’s child. In other words, spouse and child are defined to be semantically disjoint. Following the condition of disjoint axiom, we have to traverse the whole knowledge graph to find out all the relations with $s$ being the subject entity and $o$ being the object entity. The relation set is $R(s, o) = \{r_i | (s, r_i, o) \in G, r_i \neq r\}$. However, calculating the semantic compatibility of the relation pairs $(r, r_o)$ is time-consuming. We simply copy the idea from TransE, which holds the view that $s + r \approx o$ to reduce time cost. Specifically, we regard $(o_m - s_m)$ as the unified representation of relations in triples with $s$ and $o$ being the subject and object entity. Therefore, the semantic judgment of this axiom can be simplified to calculate the compatibility score of $(o_m - s_m)$ and $r_m$, which is defined as:

$$f(r_m, (o_m - s_m)) = \sigma(r_m \cdot (o_m - s_m))$$

(14)

where $\sigma$ denotes sigmoid function.

3.3.4. Irreflexive Axiom Module

Irreflexive axiom module considers two aspects of judgement. 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. Thus, only the two conditions the relation $r$ is irreflexive and $s = o$ are fulfilled simultaneously, the triple violates irreflexive axioms. For example, for the irreflexive relation 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. If $s = o$, we further consider the property of the relation. Otherwise, the probability that the triple conforms to irreflexive axiom is 1 as $s \neq o$ already violates one of the two conditions. For the case of $s = o$, we can infer that types of $s$ and $o$ are the same ($s_c = o_c$). 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 $o$. That is $r_s \approx s_c$ and $r_o \approx o_c$. Then, it can be conclude that $r_s$ and $r_o$ are compatible ($r_s \approx r_o$). We use a compatibility module to measure the type constraint, which is calculated as $\sigma(r_o \cdot r_s)$. Moreover, the semantic information of the relation $r_m$ can also help to determine the property of the relation. We utilize a multi-layer perceptron (MLP) layer to encode semantic, and domain and range of the relation. The probability that a triple satisfies irreflexive axiom is calculated through the Logic AND operation.
4. Experiments

We evaluate our proposed method NeuRAN on 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 [38] and WN18RR [18] to evaluate NeuRAN. 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 consisting billions of real-world facts. WN18 is a subset of WordNet [39] that describes relations between words.

Table 2 Statistics of FB15K237 and WN18RR

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Ent</th>
<th>#Rel</th>
<th>#Train</th>
<th>#Valid</th>
<th>#Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB15K237</td>
<td>14541</td>
<td>237</td>
<td>272115</td>
<td>17535</td>
<td>20466</td>
</tr>
<tr>
<td>WN18RR</td>
<td>40943</td>
<td>11</td>
<td>86835</td>
<td>3034</td>
<td>3134</td>
</tr>
</tbody>
</table>

Table 3 Statistics of negative triples generated from FB15K237 and WN18RR

<table>
<thead>
<tr>
<th>Datasets</th>
<th>FB15K237-10%</th>
<th>FB15K237-20%</th>
<th>FB15K237-40%</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Neg triple</td>
<td>27211</td>
<td>54423</td>
<td>108846</td>
</tr>
<tr>
<td>Datasets</td>
<td>WN18RR-10%</td>
<td>WN18RR-20%</td>
<td>WN18RR-40%</td>
</tr>
<tr>
<td>#Neg triple</td>
<td>8683</td>
<td>17367</td>
<td>34734</td>
</tr>
</tbody>
</table>

Error Imputation. Since KGs are constructed in an automated or semi-automated way, noises cannot be avoided. However, existing knowledge graph reasoning methods assume that triples in KGs are positive triples. And there are no pre-given noisy triples in FB15K237 and WN18RR. In order to verify our method, we generate new datasets with different noise rates based on the two datasets to simulate the real noisy knowledge graphs. Before generating noises, for each dataset with training, validation and test sets, we generate negative triples for the validation and test sets. The positive and negative triples in the validation set are used to find the optimum thresholds for each relation. To evaluate the performance of NeuRAN on
Neural Axiom Network for Knowledge Graph Reasoning

4.2. Noise Detection

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

Evaluation Protocol. First of all, we compute the score of the triple \((s, r, o)\) via the energy function
\[
E(s, r, o) = E_{kge} + \lambda \cdot \left[ (1 - P_{dm}) + (1 - P_{rg}) + (1 - P_{dis}) + (1 - P_{raw}) + (1 - P_{uar}) \right].
\]
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 classifies the noises as errors. Before calculating the auc metric, we normalize the energy function scores into the \([0, 1]\) interval, values close to 0 indicate correct triples, and values close to 1 indicate incorrect triples.

Result Analysis. Evaluation results on noisy datasets generated based on FB15K237 and WN18RR can

Table 4

<table>
<thead>
<tr>
<th></th>
<th>FB15K237-10%</th>
<th>FB15K237-20%</th>
<th>FB15K237-40%</th>
<th>WN18RR-10%</th>
<th>WN18RR-20%</th>
<th>WN18RR-40%</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE</td>
<td>0.9805</td>
<td>0.9799</td>
<td>0.9793</td>
<td>0.9377</td>
<td>0.9308</td>
<td>0.9291</td>
</tr>
<tr>
<td>CKRL(TransE)</td>
<td>0.9809</td>
<td>0.9803</td>
<td>0.9660</td>
<td>0.9337</td>
<td>0.9255</td>
<td>0.8971</td>
</tr>
<tr>
<td>NeuRAN(TransE)</td>
<td>0.9807</td>
<td>0.9807</td>
<td>0.9802</td>
<td>0.9403</td>
<td>0.9337</td>
<td>0.9141</td>
</tr>
<tr>
<td>TransH</td>
<td>0.9769</td>
<td>0.9752</td>
<td>0.9755</td>
<td>0.9338</td>
<td>0.9185</td>
<td>0.8778</td>
</tr>
<tr>
<td>CKRL(TransH)</td>
<td>0.9663</td>
<td>0.9697</td>
<td>0.9683</td>
<td>0.9279</td>
<td>0.9091</td>
<td>0.8527</td>
</tr>
<tr>
<td>NeuRAN(TransH)</td>
<td>0.9781</td>
<td>0.9783</td>
<td>0.9796</td>
<td>0.9340</td>
<td>0.9239</td>
<td>0.8780</td>
</tr>
</tbody>
</table>
Triple classification results on WN18RR, WN18RR-10%, WN18RR-20% and WN18RR-40%. "ACC", "P" and "R" are the abbreviation of "accuracy", "precision" and "recall", respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>WN18RR-10%</th>
<th></th>
<th></th>
<th>WN18RR-20%</th>
<th></th>
<th></th>
<th>WN18RR-40%</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>P</td>
<td>R</td>
<td>ACC</td>
<td>P</td>
<td>R</td>
<td>ACC</td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>TransE</td>
<td>0.8764</td>
<td>0.8799</td>
<td>0.8303</td>
<td>0.8857</td>
<td>0.8889</td>
<td>0.8172</td>
<td>0.8355</td>
<td>0.9046</td>
<td>0.7502</td>
</tr>
<tr>
<td>CKRL(TransE)</td>
<td>0.8789</td>
<td>0.9201</td>
<td>0.8232</td>
<td>0.8591</td>
<td>0.8973</td>
<td>0.8111</td>
<td>0.8397</td>
<td>0.8909</td>
<td>0.7741</td>
</tr>
<tr>
<td>NeuRAN(TransE)</td>
<td>0.8856</td>
<td>0.9281</td>
<td>0.8360</td>
<td>0.8703</td>
<td>0.9100</td>
<td>0.8197</td>
<td>0.8598</td>
<td>0.9328</td>
<td>0.7754</td>
</tr>
<tr>
<td>TransH</td>
<td>0.8497</td>
<td>0.9111</td>
<td>0.7750</td>
<td>0.8283</td>
<td>0.8761</td>
<td>0.7648</td>
<td>0.7921</td>
<td>0.8561</td>
<td>0.7023</td>
</tr>
<tr>
<td>CKRL(TransH)</td>
<td>0.8556</td>
<td>0.9233</td>
<td>0.7757</td>
<td>0.8403</td>
<td>0.8829</td>
<td>0.7846</td>
<td>0.8116</td>
<td>0.8629</td>
<td>0.7409</td>
</tr>
<tr>
<td>NeuRAN(TransH)</td>
<td>0.8687</td>
<td>0.9300</td>
<td>0.7974</td>
<td>0.8598</td>
<td>0.9040</td>
<td>0.8050</td>
<td>0.8323</td>
<td>0.8802</td>
<td>0.7693</td>
</tr>
</tbody>
</table>

Triple classification results for FB15K237-10%, FB15K237-20% and FB15K237-40%. "ACC", "P" and "R" are the abbreviation of "accuracy", "precision" and "recall", respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>FB15K237-10%</th>
<th></th>
<th></th>
<th>FB15K237-20%</th>
<th></th>
<th></th>
<th>FB15K237-40%</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC</td>
<td>P</td>
<td>R</td>
<td>ACC</td>
<td>P</td>
<td>R</td>
<td>ACC</td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>TransE</td>
<td>0.7802</td>
<td>0.7765</td>
<td>0.7870</td>
<td>0.7606</td>
<td>0.7496</td>
<td>0.7826</td>
<td>0.7439</td>
<td>0.7340</td>
<td>0.7650</td>
</tr>
<tr>
<td>CKRL(TransE)</td>
<td>0.7758</td>
<td>0.7743</td>
<td>0.7786</td>
<td>0.7605</td>
<td>0.7353</td>
<td>0.8140</td>
<td>0.7422</td>
<td>0.7269</td>
<td>0.7759</td>
</tr>
<tr>
<td>NeuRAN(TransE)</td>
<td>0.7810</td>
<td>0.7846</td>
<td>0.7749</td>
<td>0.7656</td>
<td>0.7711</td>
<td>0.7554</td>
<td>0.7484</td>
<td>0.7497</td>
<td>0.7459</td>
</tr>
<tr>
<td>TransH</td>
<td>0.7969</td>
<td>0.8055</td>
<td>0.7826</td>
<td>0.7800</td>
<td>0.7877</td>
<td>0.7666</td>
<td>0.7646</td>
<td>0.7624</td>
<td>0.7687</td>
</tr>
<tr>
<td>CKRL(TransH)</td>
<td>0.7978</td>
<td>0.8023</td>
<td>0.7904</td>
<td>0.7831</td>
<td>0.7816</td>
<td>0.7857</td>
<td>0.7623</td>
<td>0.7754</td>
<td>0.7387</td>
</tr>
<tr>
<td>NeuRAN(TransH)</td>
<td>0.7882</td>
<td>0.8080</td>
<td>0.7561</td>
<td>0.7784</td>
<td>0.7891</td>
<td>0.7600</td>
<td>0.7617</td>
<td>0.7796</td>
<td>0.7299</td>
</tr>
</tbody>
</table>

4.3. Triple Classification

Triple classification aims to judge whether a triple in the test set is correct or not, according to triple scores calculated by the energy function $E(s, r, o) = E_{\text{edge}} + A \cdot [\{(1 - P_{\text{DM}}) + (1 - P_{\text{IRRE}}) + (1 - P_{\text{DIS}}) + (1 - P_{\text{ASYM}})\}]$, which can be viewed as a binary classification task on the test set.

Evaluation Protocol. As the test sets of the datasets used for triple classification only have correct triples, we generate negative triples by corrupting the subject or object entity of correct triples randomly. For the validation and test sets, the number of negative triples is the same as the number of positive triples. Thus there are labeled positive and negative triples in the two sets. For example, for WN18RR-based datasets, the number of triples is 6068 in the validation set, and is 6268 in the test set. For triple classification, we learn a relation-specific threshold $\delta_r$ for every relation. $\delta_r$ 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 $\delta_r$, it is classified as positive, otherwise negative. We use accuracy(ACC), precision(P) and recall(R) as the evaluation metrics.
Table 7
Link prediction results on FB15K237-10%, FB15K237-20% and FB15K237-40%.

<table>
<thead>
<tr>
<th>Model</th>
<th>MRR</th>
<th>Hit@1</th>
<th>MRR</th>
<th>Hit@1</th>
<th>MRR</th>
<th>Hit@1</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE</td>
<td>0.258</td>
<td>0.299</td>
<td>0.159</td>
<td>0.240</td>
<td>0.279</td>
<td>0.144</td>
</tr>
<tr>
<td>CKRL(TransE)</td>
<td>0.252</td>
<td>0.294</td>
<td>0.150</td>
<td>0.236</td>
<td>0.278</td>
<td>0.136</td>
</tr>
<tr>
<td>NeuRAN(TransE)</td>
<td>0.284</td>
<td>0.313</td>
<td>0.199</td>
<td>0.269</td>
<td>0.292</td>
<td>0.189</td>
</tr>
<tr>
<td>TransH</td>
<td>0.241</td>
<td>0.296</td>
<td>0.125</td>
<td>0.213</td>
<td>0.270</td>
<td>0.094</td>
</tr>
<tr>
<td>CKRL(TransH)</td>
<td>0.194</td>
<td>0.254</td>
<td>0.070</td>
<td>0.172</td>
<td>0.229</td>
<td>0.050</td>
</tr>
<tr>
<td>NeuRAN(TransH)</td>
<td>0.288</td>
<td>0.317</td>
<td>0.203</td>
<td>0.270</td>
<td>0.293</td>
<td>0.189</td>
</tr>
</tbody>
</table>

Table 8
Link prediction results on WN18RR-10%, WN18RR-20% and WN18RR-40%.

<table>
<thead>
<tr>
<th>Model</th>
<th>MRR</th>
<th>Hit@1</th>
<th>MRR</th>
<th>Hit@1</th>
<th>MRR</th>
<th>Hit@1</th>
</tr>
</thead>
<tbody>
<tr>
<td>TransE</td>
<td>0.211</td>
<td>0.351</td>
<td>0.032</td>
<td>0.206</td>
<td>0.349</td>
<td>0.031</td>
</tr>
<tr>
<td>CKRL(TransE)</td>
<td>0.215</td>
<td>0.350</td>
<td>0.044</td>
<td>0.205</td>
<td>0.340</td>
<td>0.034</td>
</tr>
<tr>
<td>NeuRAN(TransE)</td>
<td>0.342</td>
<td>0.393</td>
<td>0.256</td>
<td>0.334</td>
<td>0.392</td>
<td>0.247</td>
</tr>
<tr>
<td>TransH</td>
<td>0.185</td>
<td>0.290</td>
<td>0.039</td>
<td>0.173</td>
<td>0.276</td>
<td>0.032</td>
</tr>
<tr>
<td>CKRL(TransH)</td>
<td>0.184</td>
<td>0.288</td>
<td>0.040</td>
<td>0.174</td>
<td>0.277</td>
<td>0.032</td>
</tr>
<tr>
<td>NeuRAN(TransH)</td>
<td>0.330</td>
<td>0.380</td>
<td>0.251</td>
<td>0.328</td>
<td>0.378</td>
<td>0.250</td>
</tr>
</tbody>
</table>

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 metrics, our method outperforms baselines on the WN18RR-based datasets and achieves the best results. It confirms that learning knowledge representations with axiom information can help triple classification. (2) On FB15K237-based dataset, the results are comparable with baselines. The improvements on WN18RR-10%, WN18RR-20% and WN18RR-40% is more obvious than on FB15K237-10%, FB15K237-20% and FB15K237-40%. It demonstrates that implicit axiom information is more effective on a dataset with smaller relations and triples. (3) Compared with TransE, TransH, CKRL(TransE) and CKRL(TransH), the higher the noise rate, the smaller the decrease in the accuracy metric of our method on WN18RR-10%, WN18RR-20% and WN18RR-40%. It indicates that on noisy datasets, triple classification results of NeuRAN can be more robust than baselines on small datasets.

From the triple classification results, we can conclude that the combination of implicit axiom and structural information reflected by existing triples in knowledge graphs works better than using only structural information on datasets with a small number of relations and triples. But path information is more useful when the number of relations and triples is large.

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 graph completion. This task aims to predict the missing entity when given one entity and one relation of a triple, including subject entity prediction ($s, r, ?$) and object 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 $e \in E$ in the dataset as candidate predictions, and then replace the missing part with each entity $e$ and calculate scores for the triples in $T = \{(e, r, o) | e \in G \}$. 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 the 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 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 on all metrics. The improvements are significantly on all metrics, especially on Hit@1. It demonstrates that axiom information is of great help in improving the predictive ability of a missing triple, when the 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. 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 datasets.

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.

### 4.5. Ablation study

We conduct ablation studies on link prediction to assess the effectiveness of NeuRAN. As our model is composed of a knowledge graph embedding module and five neural axiom modules, we add each axiom module to the knowledge graph embedding module to investigate the contributions of the axiom module. Specifically, we use the score function and loss function defined in equation 1 and 2 to train our model. TransE is taken as the knowledge graph embedding module. For evaluation, we set the score function as $E(s, r, o) = E_{kge} + \lambda \cdot E_a$ to illustrate the impact of each axiom module. $E(s, r, o)$ is the score of the triple, and $E_{kge}$ is the score from the knowledge graph embedding module. $E_a$ means the score of the selected axiom, and can be $(1 - P_{dm}), (1 - P_{eg}), (1 - P_{dis}), (1 - P_{ser})$ or $(1 - P_{dis})$.

From Table 9, we can observe that adding any of these five axioms can improve link prediction results on WN18RR-10%. The disjoint and irreflexive modules work better than other modules. Particularly, the irreflexive axiom module has shown substantial improvements on MRR and Hit@1 metrics. As the number of relations is small and most relations are asymmetric on WN18RR%, using attention mechanism to aggregate relation representations or adding the asym-

<table>
<thead>
<tr>
<th></th>
<th>WN18RR-10%</th>
<th>FB15K237-10%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRR</td>
<td>Hit@</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>KGE</td>
<td>0.2103</td>
<td>0.3468</td>
</tr>
<tr>
<td>KGE+DM</td>
<td>0.2108</td>
<td>0.3476</td>
</tr>
<tr>
<td>KGE+RG</td>
<td>0.2134</td>
<td>0.3500</td>
</tr>
<tr>
<td>KGE+DIS</td>
<td>0.2203</td>
<td>0.3529</td>
</tr>
<tr>
<td>KGE+IRRE</td>
<td>0.3407</td>
<td>0.3925</td>
</tr>
<tr>
<td>KGE+ASYM</td>
<td>0.2108</td>
<td>0.3472</td>
</tr>
<tr>
<td>KGE+ALL</td>
<td>0.3417</td>
<td>0.3926</td>
</tr>
</tbody>
</table>
metric module have a small gain. For the results on FB15K237-10% dataset with more relations, noisy triples can affect the aggregated representations. Besides, the small number of asymmetric relations makes the asymmetric module ineffective. Although the domain, range and asymmetric modules are not as efficient as the other modules, we consider them for a comprehensive exploration.

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 to enhance the performance of knowledge graph reasoning. Furthermore, we will improve our method to apply it for inconsistency reasoning, as axiom information may be able to provide explanations for inconsistent triples.

References


[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.


