

# Logic Diffusion for Knowledge Graph Reasoning

Xiaoying Xie\*  
xiaoyingx@stu.xjtu.edu.cn  
Xi'an Jiaotong University

Biao Gong\*  
Yiliang Lv  
a.biao.gong@gmail.com  
401851090@qq.com

Zhen Han  
hanzhn@qq.com  
Xi'an Jiaotong University

Guoshuai Zhao  
guoshuai.zhao@xjtu.edu.cn  
Xi'an Jiaotong University

Xueming Qian  
qianxm@mail.xjtu.edu.cn  
Xi'an Jiaotong University

## ABSTRACT

Most recent works focus on answering first order logical queries to explore the knowledge graph reasoning via multi-hop logic predictions. However, existing reasoning models are limited by the circumscribed logical paradigms of training samples, which leads to a weak generalization of unseen logic. To address these issues, we propose a plug-in module called *Logic Diffusion (LoD)* to discover unseen queries from surroundings and achieves dynamical equilibrium between different kinds of patterns. The basic idea of *LoD* is relation diffusion and sampling sub-logic by random walking as well as a special training mechanism called gradient adaption. Besides, *LoD* is accompanied by a novel loss function to further achieve the robust logical diffusion when facing noisy data in training or testing sets. Extensive experiments on four public datasets demonstrate the superiority of mainstream knowledge graph reasoning models with *LoD* over state-of-the-art. Moreover, our ablation study proves the general effectiveness of *LoD* on the noise-rich knowledge graph.

## KEYWORDS

knowledge graph reasoning, multi-hop logic reasoning, First-Order-Logic

## 1 INTRODUCTION

Knowledge graph (KG) provides a structural data representation which is organized as triples of entity pairs and relationships [3, 47, 54]. In recent year, KG-based common sense reasoning algorithms usually combine mathematical logic [20, 63], relational path [25, 29, 46], distributional representation [43, 48, 62], etc. [1, 21, 30] with deep learning models to answer First-Order-Logical (FOL) queries, which greatly enhanced reasoning performance and achieved generalization. Different from the basic structural triplets like  $(e_s, r, e_o)$ , FOL implements logic by existential quantification ( $\exists$ ), conjunction ( $\wedge$ ), disjunction ( $\vee$ ), and negation ( $\neg$ ), which is well suited for describing relationships. Around such logical units, the learning objective of models usually focuses on logical mappings rather than representations to achieve the best logical answers. However, one of the limitations of this former is that the logical paradigms need to be defined manually (e.g., [22, 59]). Also, as mentioned above, since the weak ability of learning representations, the impact of unseen FOL and noisy data with unreliable logic on reasoning performance is catastrophic. Coincidentally, these two elements are abundant and inevitable in the real-world knowledge graph [19, 27, 56].

\*Both authors contributed equally to this research.

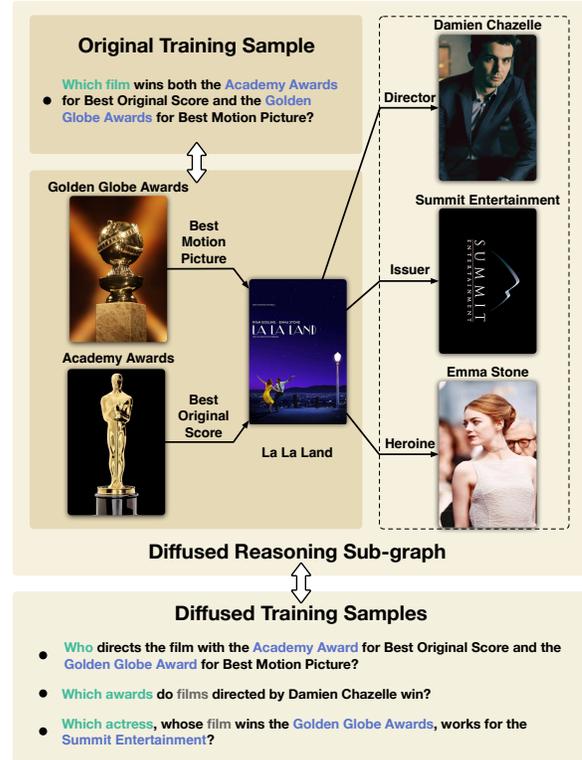


Figure 1: Illustration of original training samples and extended training samples after the diffusion. How to discover unseen paradigms from seen ones is what *LoD* tries to solve.

Based on the above thinking, we summarize two major problems in the real-world knowledge graph: (1) discovering unseen FOL paradigms, and (2) learning non-duality logic. Specifically, the unseen FOL refers to the FOL that does not appear in the training data but needs to answer during testing. While reasoning models have a limited ability to learn these queries, it is implicit in the training process and leads to a severe impairment of performance. Therefore, it is beneficial to be proactive in discovering unseen FOL during training. Secondly, the non-duality logic means the unreliable inverse mapping between entities. The inverse self-consistency of the logic is very rare in the noise-rich knowledge graph, and by learning this inverse mapping which we called non-duality logic, the model can achieve very strong robustness.

Recent studies (e.g., [32]) have divided the seen / unseen logical paradigms to fairly evaluate models and [8, 37, 40, 51] has used variational inference to handle potential noise in real-world knowledge graphs, none of them analyzed the implications of making these attempts on noisy knowledge graph in a holistic way. As a result, there is no holistic solution that can be specifically used to solve robust common sense reasoning problems on unseen logical paradigms. In this paper, we consider a plug-in module called *Logic Diffusion (LoD)* to address these issues.

*LoD* relies on a new structure called *Hierarchical Conjunctive Query* which achieves a wide range of logic perception through different levels of unseen FOL discovery. Then, with the help of *Logic Specific Prompt*, *LoD* can distinguish between different logical paradigms as well as learn the commonalities of the same kind of FOL. However, due to the difference in the learning difficulty, the model tends to learn dominant or simple FOL, which is detrimental to the overall performance. Thus, we design a training mechanism called *Gradient Adaption* in *LoD* to slightly suppress the gradient on the fastest converging FOL and make the model have more opportunities to learn difficult or long-tailed samples in the early stage of training. Figure 1 is the illustration of the difference in FOL with or without *LoD*.

Besides, *LoD* is accompanied by a novel loss function to further achieve the robust logical diffusion when facing noisy data in training or testing sets. The key idea of our loss function is extending the length of the mapping link and amplifying the perturbation, so that the noise data can be blocked. Following [19, 51], we divide the noisy data of non-duality logic into (1) training with noise, and (2) testing with noise. The model with our loss function tends to retain more FOL inputs that can be successfully reverse mapped in the training phase, and actively eliminates FOL inputs that cannot be successfully reverse mapped in the testing phase. Such a process ensures that the model has the ability to learn reverse mapping and remembers the useful data. We use a mixed dataset to simulate what contains massive noise data of non-duality logic. In Section 5, we did detailed noise ratio experiments based on two commonly used public datasets FB15K [3] and NELL-995 [54].

In summary, the main contributions of our work are three-fold:

- We propose a plug-in module called *Logic Diffusion (LoD)* which is extremely effective reasoning on both unseen and seen logical paradigms. To the best of our knowledge, *LoD* is the first work to focus on logic diversity augmentation on logic perspective in knowledge graph reasoning by a flexible module.
- *LoD* is accompanied by a novel loss function to further achieve the robust logical diffusion when facing noisy data in training or testing sets.
- Extensive experiments on four public datasets demonstrate the superiority of mainstream KG reasoning models with the proposed *LoD* plug-in module over state-of-the-art. Moreover, our ablation study proves the general effectiveness of *LoD* on the noise-rich knowledge graph.

## 2 RELATED WORK

Logical query reasoning on knowledge graphs has been recently received growing interest. Generally, this work contains

two main lines of works: how to model multi-hop relations and how to model numerous answers [11, 58].

### 2.1 Modeling Multi-hop Relations

Multi-hop logic reasoning try to answer queries with multi-hop logic permutations. Since embedding entities and relations in knowledge graph (KG) into low-dimensional vector space has been widely studied. Various works [5, 6, 28, 33, 34, 41, 49, 57] can answer single-hop relational queries via link prediction but these models cannot handle complex logical reasoning. Therefore, to answer multi-hop FOL queries[15], Graph Query Embedding (GQE) [17] encodes conjunctive queries through a computation graph with relational projection and conjunction ( $\wedge$ ) as operators. While path-based (i.e., deep reinforcement learning based) methods [7, 25, 26, 29, 46, 52–54] start from anchor entities and determine the answer set by traversing the intermediate entities via relational path and graph neural convolution based methods [18, 23, 36, 38, 55, 60] pass message to iterate graph representation for reasoning.

### 2.2 Modeling Numerous Answers

Traditional knowledge graph reasoning works do not pay attention to potentially large sets of answer entities [31]. That is, as long as one correct answer is inferred, KGR models is considered valid. However, it is unclear how such an entity set containing numerous answers can be represented as a single point in the vector space, causing inference inconsistent with the real situation. So in order to handle numerous answer entities and inspired by metric learning, a series of works embeds queries into geometric shapes [17, 31, 61], probability distributions [12, 13, 21, 24, 50], and complex objects [2, 14, 42]. Then by optimizing the similarity metrics between answer entities and queries, entities within border distance metrics of various representation spaces are regarded as correct answers.

However, above works lack of generalization to modeling queries of unseen logical paradigms and suffer from the interference of noisy data with non-duality logic, which are unavoidable in complex logical reasoning tasks on real-world knowledge graphs.

## 3 PRELIMINARIES

### 3.1 Logic Format

In the field of knowledge graph reasoning, most recent works focus on answering First-Order-logical queries rather than single-hop traversal within the triplet level. It is because answering FOL queries requires proper representation in both embedding and logic perspectives. As it is illustrated in Figure 1, the query “Who directs the film with the Academy Award for Best Original Score and the Golden Globe Award for Best Motion Picture?” can be structured as a reasoning sub-graph  $G_q$  [15]. Entities “Golden Globe Award” and “Academy Award” are anchor entities, while entity “Damien Chazelle” is the target entity which refers to the answer, consisting a specific logical pattern of FOL.

More specifically, First-Order Logic (FOL) is logic paradigms consisting of logical operators as conjunction ( $\wedge$ ), disjunction ( $\vee$ ), universal quantification ( $\forall$ ), existential quantification ( $\exists$ ) and negation ( $\neg$ )<sup>1</sup>. Structured query-answer pairs of different FOL

<sup>1</sup>Note that queries with universal quantification do not apply in real-world knowledge graphs since no entity connects with all the other entities. Furthermore, if it is necessary

paradigms are the input of knowledge graph reasoning models during training while only queries during testing. Apparently it is impossible to exhaust all logical patterns. Moreover, in order to evaluate the generalization ability to unseen paradigms, paradigms for inference are more than training ones. While FOL paradigms without negation operators are Existential-Positive-First-order (EPFO), focused by some other works.

### 3.2 Knowledge Graph Embedding

Before logical reasoning, knowledge graph embedding (KGE) needs to map entities and relations in KG onto a representational latent space, which can participate in the former logical operations. Given a set of entities  $\mathcal{E}$  and a set of relations  $\mathcal{R}$ , a knowledge graph  $\mathcal{G} = \{(e_s, r, e_o)\} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$  consists of factual triples as subject  $e_s \in \mathcal{E}$ , object  $e_o \in \mathcal{E}$  and relational functions  $r \in \mathcal{R} : \mathcal{E} \times \mathcal{E} \rightarrow \{\text{True}, \text{False}\}$  or other confidence and distributional metrics. Suppose  $e_s \in \mathbb{R}^d$ ,  $e_o \in \mathbb{R}^d$ ,  $r \in \mathbb{R}^d$  are vector representations of subject  $e_s$ , object  $e_o$  and relation  $r$  in a triple of knowledge graphs, KGE works usually optimize their models according to a relation projection function  $e_o = f_r(e_s)$ . As a result, the embedding features can be extracted from the embedding layer of pre-trained KGE models to computing the former logical operations.

### 3.3 Knowledge Graph Reasoning

As mentioned above, answering a FOL query  $q$  can be simply illustrated as finding its answer set  $\llbracket q \rrbracket$  according to its reasoning sub-graph  $G_q$ . A reasoning sub-graph is an abbreviation of the computation graph where nodes refer to entity sets and edges refer to logical operations. We call the starting point of FOL as anchor entities, and the end point of FOL as target entities.

Thus we can answer  $q$  by executing logical operators from anchor entities. Based on this premise, logical operators can be matched according to the following rules:

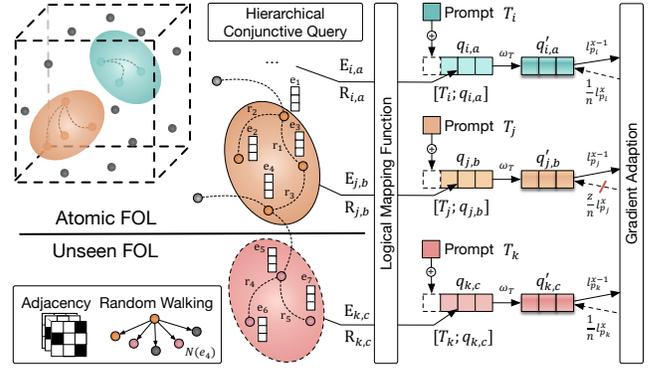
- *Relation Projection* : given a set  $S \subseteq \mathcal{E}$  of entities and relation operator  $r \in \mathcal{R}$ , compute entities  $\cup_{e \in S} P_r(e)$  adjacent to  $S$  via  $r$ :  $P_r(e) \equiv \{e' \in \mathcal{E} : r(e, e') = \text{True}\}$ .
- *Intersection* : Given  $n$  sets  $\{S_1, S_2, \dots, S_n\}$  of entities, compute their intersection  $\cap_{i=1}^n S_i$ .
- *Union* : Given  $n$  sets  $\{S_1, S_2, \dots, S_n\}$  of entities, compute their union  $\cup_{i=1}^n S_i$ .
- *Negation* : Given a set  $S \subseteq \mathcal{E}$  of entities, compute its complement  $\bar{S} \equiv \mathcal{E} \setminus S$ .

In a word, knowledge graph reasoning mainly aims at answering FOL queries by executing several logical operators with vector representations after embedding.

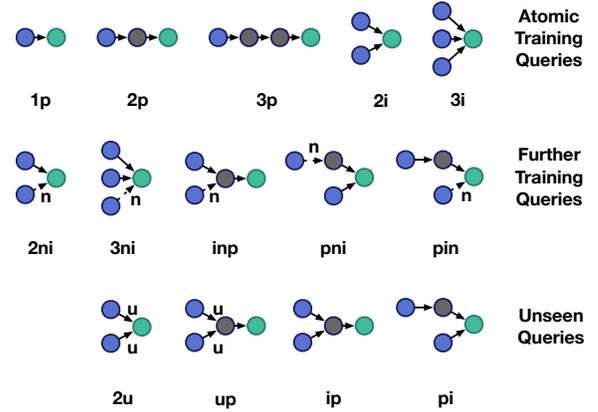
## 4 LOGIC DIFFUSION

In this section, we introduce the proposed plug-in module—*Logic Diffusion* in detail. *LoD* accepts any permutation and combination of FOL instances of clean or noise-rich KG as input. Referring to Figure 2 and Figure 3, along the direction of data flow, *LoD* has two main designs including *LoD* architecture and adaptive loss. *LoD* architecture, consisting of [Hierarchical Conjunctive Query],

to introduce the universal quantifier into KG reasoning, the universal quantifier can be transformed from the existential quantification and the negation. Thus we will not discuss the queries with the universal quantifier.



**Figure 2: Overall framework Logic Diffusion architecture.** Along the direction of data flow, there are Hierarchical Conjunctive Query, Logic Specific Prompt and Gradient Adaption, discovering and learning multiple FOL paradigms.

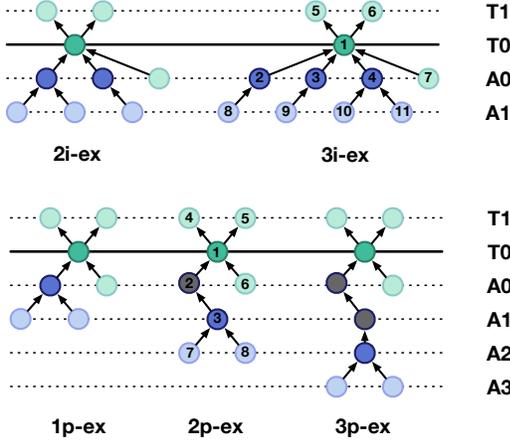


**Figure 3: Illustration of 14 typical FOL paradigms, including atomic training, further training and unseen queries.**

[Logic Specific Prompt] and [Gradient Adaption] is used to discover and learn multiple FOL paradigms. Note that the logic diffusion process mainly happens during [Hierarchical Conjunctive Query], which extends logical diversity by random walking among the entity distribution; [Logic Specific Prompt] specializes the feature within each FOL paradigms while distinguishes among different ones; [Gradient Adaption] dynamically adjusts convergence speed and achieves adaptively balance of several logic patterns. While *LoD Loss* is used to keep duality logic during robust reasoning and release the interference of noisy data in real-world KG.

### 4.1 LoD Architecture

**Hierarchical Conjunctive Query.** In this section, we introduce Hierarchical Conjunctive Query, which is the core component of logical diffusion, discovering unseen logic paradigms and sample sub-logic patterns both in abstract reasoning sub-graphs and figurative instances. To implement Logic Diffusion, we firstly define Hierarchical Conjunctive Query as a theoretical guide. Following



**Figure 4: Illustration of the Hierarchical Conjunctive Query. Each group represents a FOL paradigm. Nodes T0 layer contains the positive target entity  $t$  which indicates the answer of a certain FOL.  $e_i$  is one layer below  $e_j$  if there exists  $(e_i, e_j)$  satisfying  $r(e_i, e_j) = \text{True}$  where  $r$  is the relation mapping.**

[32], we selected 14 typical FOL paradigms illustrated in Figure 3 where atomic training queries and further training queries denote the input samples of KGR models, while unseen queries denote samples do not participate training process but evaluation. Though there are far more FOL paradigms, we can still learn the generalization ability of KGR models through modeling these unseen queries in this specific setting. As the name suggests, Hierarchical Conjunctive Query has a hierarchical form, from the bottom to top which is:

- *Bottom Layer* : Atomic Query

Following [32], there are 14 typical FOL paradigms including atomic training  $\{1p, 2p, 3p, 2i, 3i\}$ , further training  $\{2ni, 3ni, inp, pni, pin\}$  and unseen  $\{ip, pi, 2u, up\}$ . Since the atomic training queries  $\{1p, 2p, 3p, 2i, 3i\}$  are raw and simple input samples, we simply take them as the bottom layer of Hierarchical Conjunctive Query.

- *Middle Layer* : Sub-Graph Diffusion

The middle layer of Hierarchical Conjunctive Query contains the extensions and permutations of Atomic Queries, which is theoretically infinite (e.g.,  $\{9p, 10i, 7i3p \dots\}$ ). As shown in Figure 4, given a query  $q$ , reasoning sub-graph  $G_q$  and positive target  $t$  of an atomic query, we diffuse  $G_q$  by neighbors. Suppose  $N(e) \equiv \{e_{neib} | r(e_{neib}, e) \vee r(e, e_{neib}), r \in \mathcal{R}\}$  is the collection of neighbors of each central node  $e$ , we get  $N(e_1), N(e_2), \dots, N(e_n)$  and  $N(t)$ . If  $r(e_{neib}^j, e_i) = \text{True}$  or  $t_{neib}^j \in N(t)$ , we associate  $e_{neib}^j$  and  $t_{neib}^j$  to  $G_q$  via  $r$ . We stipulate that  $e_{neib}^j$  is different from any other entities in  $q$  and  $t$ .

- *Top Layer* : Unseen Query

The top layer contains unseen FOL which are not present in the training set. It is generated by random walking in the diffused reasoning sub-graphs to cover complex logic in the real-world knowledge graph, i.e., they are not given directly with the training set. Here we give examples of  $3i - ex$  and  $2p - ex$ . From  $3i - ex$ , we

can retrieve a sub-structure (nodes 1, 2, 3, 5 and relational edges attached, etc.) corresponding to  $ic$ . From  $2p - ex$ , we can retrieve a sub-structure (nodes 1, 2, 3, 6 and relational edges attached, etc.) corresponding to  $ci$ . Note that each node in the diagram actually represents a collection of entities, which is only represented as a single entity to simplified.

In the paradigm  $3i - ex$ , for  $ic$  query with nodes 1, 2, 3, 5, we label nodes 2, 3 as the new anchor entities, and node 5 as the new positive target entity. Then we randomly sample  $n_s$  entities  $\notin N(1)$  as the negative target entities where  $N(1)$  denotes the collection of neighbors of node 1.

In paradigm  $2p - ex$ , for  $ci$  query with nodes 1, 2, 3, 6, note since node 2 is unknown to models, we will not choose this kind of nodes as the new target nor anchor entities. Similarly we label nodes 2, 6 as the new anchor entities, node 5 as the new positive target entities, and ones  $\notin N(6)$  as the negative target entities.

**Logic Specific Prompt.** Stemming from recent advances in natural language processing, prompt learning initially fills the input sample into properly handcrafted prompt templates, so that a pre-trained language model can “understand” the task [4]. Similarly, we define different kinds of FOL as different tasks and propose Logic Specific Prompt to make the learning of the model more targeted. Specifically, given a set  $\{Q_1, Q_2, \dots, Q_k\}$  belonging to  $k$  kinds of FOL, we generate  $k$  random vectors  $\{T_1, T_2, \dots, T_k\}$  from normal distribution as initialized prompt. Then we concatenate prompt  $T_i$  with corresponding  $q_{i,j}$  as  $q'_{i,j}$ . The formula of  $q'_{i,j}$  is as follow:

$$q'_{i,j} = \text{ReLU}(\omega_T [T_i; q_{i,j}] + b_T) \quad (1)$$

where  $T_i \in \mathbb{R}^{d_T}$  in which  $d_T$  denotes the dimension of prompt, filter  $\omega_T \in \mathbb{R}^{(d+d_T) \times d}$ ,  $b_T \in \mathbb{R}^d$  is the bias. The optimal value of  $d_T$  is 32.

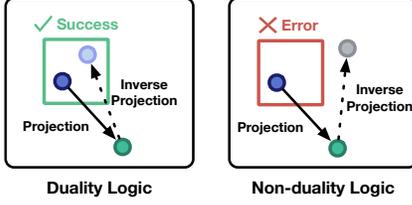
**Gradient Adaption.** Our approach involves a rich variety of FOL due to the introduction of Hierarchical Conjunctive Query. If no restrictions are imposed, the model will tend to learn dominant or simple FOL, which is detrimental to the overall performance. The key idea of Gradient Adaption is to slightly suppress the gradient on the fastest converging FOL. Through this processing, the model has more opportunities to learn difficult or long-tailed samples in the early stage of training. Specifically, suppose  $l_p^{i-1} = \{l_{p_1}^{i-1}, l_{p_2}^{i-1}, \dots, l_{p_n}^{i-1}\}$  is the calculated loss value of  $\{p_1, p_2, \dots, p_n\}$  at the  $(i-1)$ -th iteration,  $l_p^i = \{l_{p_1}^i, l_{p_2}^i, \dots, l_{p_n}^i\}$  at the  $i$ -th iteration. The suppression of  $l_p^i$  is computed as follow:

$$l_p^{i'} = \frac{1}{n} \left( \sum_{j=1}^n l_{p_j}^i - z \cdot l_{p_\zeta}^i \right) \quad (2)$$

The attenuation factor  $z$  is 0.05.  $\zeta = \underset{n \in \mathcal{R}}{\text{argmax}} l_{p_n}^{i-1}$ .

## 4.2 LoD Loss

In retrospect, we disassemble the noise-rich knowledge graph into two major problems: unseen logic paradigms and noisy data with non-duality logic. In this section, we will introduce our solution for non-duality logic. As introduced in Section 3, a complex query constructed with permutation of logic operators can be simply illustrated as a reasoning sub-graph  $G_q$ . Answer queries by executing logic operators from anchor entities to target entities is defined as



**Figure 5: Logic Diffusion Loss can perfect duality logic (i.e., bidirectional mapping) between queries and answers in noise-rich KG. Thus non-duality noisy data can be distinguished by a distance threshold.**

a forward reasoning procedure. However, the existing methods use only forward procedure which leads to unreliable bidirectional mapping between queries and answers, which is non-duality. Indeed, how to design a backward reasoning procedure and perfect the inference-inversion process are the keys to solve this problem. That is, to keep duality logic in knowledge graph reasoning. Specifically, given an anchor entity set  $\{a_1, a_2, \dots, a_s\}$  of an FOL query  $q$ , we define positive / negative forward relational paths  $\varrho / \varrho'_i$  in the reasoning sub-graph  $G_q$ .  $\varrho$  starts from a certain anchor entity  $a_j$  to a positive target entity  $t \in \llbracket q \rrbracket$  through relation mappings.  $\varrho'_i$  starts from  $a_j$  to a random negative entity  $t'_i \notin \llbracket q \rrbracket$ . Apparently, the only kind of logic operators along a positive / negative forward relational path is the relation projection  $P_r(e) \equiv \{e' \in : r(e, e') = \text{True}\}$ . After defining forward relational paths, we generate backward relational paths with a similar strategy. We replace all relation projection operators in a forward relational path with their inverse mapping  $P_r^{-1}(e) \equiv \{e' \in : r(e', e) = \text{True}\}$ .  $\varrho_j^{-1}$  denotes a positive backward relational path from  $t$  to  $a_j$ .  $\varrho^{-1}_{i,j}$  denotes a negative backward relational path from  $t'_i$  to  $a_j$ .  $\mathcal{L}_{dl}^j$  of the anchor entity  $a_j$  is:

$$\mathcal{L}_{dl}^j = -\log\sigma(\text{score}(a_j, \varrho^{-1}_j)) - \sum_{i=1}^{n_s} \log\sigma(-\text{score}(a_j, \varrho^{-1}_{i,j})) \quad (3)$$

Accordingly, we define *LoD Loss* as follows:

$$\mathcal{L}_{dl} = \begin{cases} \text{Max}(\mathcal{L}_{dl}^1, \mathcal{L}_{dl}^2, \mathcal{L}_{dl}^3), & \text{if } p \in \{3i\} \\ \text{Max}(\mathcal{L}_{dl}^1, \mathcal{L}_{dl}^2), & \text{if } p \in \{2i, ip, pi\} \\ \text{Min}(\mathcal{L}_{dl}^1, \mathcal{L}_{dl}^2), & \text{if } p \in \{2u, up\} \\ \mathcal{L}_{dl}^1, & \text{if } p \in \{1p, 2p, 3p\} \end{cases} \quad (4)$$

The noisy data with non-duality logic can be distinguished by  $\text{score}(\varrho^{-1}, a)$  with the distance threshold  $\varsigma$ .

In addition to  $\mathcal{L}_{dl}$ , we also need to use a contrast loss  $\mathcal{L}_r$  to optimize the model by pulling  $(q, t)$  closer while pushing  $(q, t'_i)$  farther ( $q$  is query,  $t \in \llbracket q \rrbracket$  is target and  $t'_i \notin \llbracket q \rrbracket$  is random negative target). The formula is as follow:

$$\mathcal{L}_r = -\log\sigma(\text{score}(t, q)) - \sum_{i=1}^{n_s} \log\sigma(-\text{score}(t'_i, q)) \quad (5)$$

**Table 1: Statistics of datasets as well as training, validation and test edge splits. *Entity* means the number of entities. *Rela.* means the number of relations. *Tr-Edge* means the number of edges in training set. *V-Edge* means the number of edges in validation set. *Ts-Edge* means the number of edges in testing set. *Ts-Edge* means the number all edges.**

Dataset	Entity	Rela.	Tr-Edge	V-Edge	Ts-Edge	Edge
FB15k	14,951	1,345	483,142	50,000	59,071	592,213
FB15k-237	14,505	237	272,115	17,526	20,438	310,079
NELL995	63,361	200	114,213	14,324	14,267	142,804
WN18	40,493	18	141,442	5,000	5,000	151,442

where score denotes a normalized score function to evaluate the similarity between an entity and a query.  $\sigma$  is the sigmoid function, and  $n_s$  denotes the negative sample size. Therefore, the overall loss function  $\mathcal{L}$  can be calculated as follow:

$$\mathcal{L} = \mathcal{L}_r + \lambda \mathcal{L}_{dl} \quad (6)$$

where  $\lambda \in [0, 1]$  is a hyperparameter. The best is 0.8.

**Table 2: The best hyperparameters of *LoD*.**

$d$	lr	$b_z$	$n_s$	$M$	$d_T$	$z$
768	0.0005	512	128	64	32	0.05

## 5 EXPERIMENTS

### 5.1 Evaluation Setup

**Datasets.** We evaluate our approach over 4 standard KG datasets including **FB15k** [3], **FB15k-237** [47], **NELL995** [54] and **WN18** [3] with their with official training / validation / test edge splits shown in Table 1 containing 14 types of queries, which are created by the query construction method in [32]. Following [32], we train models which can not handle negation with queries of 1p, 2p, 3p, 2i, 3i patterns and evaluate those with all EPFO. While FOL models over atomic queries plus 2ni, 3ni, inp, pni, pin, and evaluate those over all FOL patterns. Note that BetaE [32] generates queries with answers less than a threshold, while GQE [17] and Query2Box [31] do not limit answers. We evaluate these methods following BetaE for fair comparison. Besides, as for constructing queries of noisy knowledge graphs, we randomly mix training / validation / test of FB15k and NELL995 up with different proportions respectively.

**Evaluation Metrics.** Following [32], for each non-trivial answer  $t$  of test query  $q$ , we rank it against non-answer entities  $\mathcal{E} \setminus \llbracket q \rrbracket_{\text{test}}$  [3]. Then the rank of each answer is labeled as  $r$ . We use **Mean Reciprocal Rank (MRR)**:  $\frac{1}{r}$  and **Hits at N (H@N)**:  $1[r \leq N]$  as quantitative metrics.

**Baselines.** We consider five metric learning based KG reasoning models over incomplete KG: **GQE** [17], **Query2Box** [31], **ABIN** [45], **BetaE** [32] and **FuzzQE** [10]. Note that GQE, Query2Box and ABIN cannot handle the negation operation, so we only evaluate these three methods by EPFO which does not contain the negation operation compared to FOL. All comparison methods are reproduced by the released source code except FuzzQE. As reported in the GitHub project, it usually takes four days to a week to finish a

**Table 3: MRR results (%) on answering FOL queries of raw methods and their enhanced versions (with *LoD*) over FB15k. *avg* denotes the the average MRR on all queries (i.e., FOL), *avg<sub>p</sub>* on EPFO, and *avg<sub>unseen</sub>* on unseen paradigms. Note that limited by training time, we manually reproduce the Logical Mapping Function of FuzzQE as it was described in the paper and marked as FuzzQE\*.**

Model	<i>LoD</i>	<i>avg</i>	<i>avg<sub>p</sub></i>	<i>avg<sub>unseen</sub></i>	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
FB15k																		
GQE	✗	-	28.0	20.0	55.3	15.2	11.2	39.1	51.0	27.7	18.8	22.1	11.5	-	-	-	-	-
	✓	-	<b>30.6</b>	<b>22.8</b>	57.9	16.6	13.7	42.3	54.1	31.6	21.3	24.8	13.4	-	-	-	-	-
Query2Box	✗	-	37.9	29.2	67.2	21.7	14.3	54.9	66.3	39.7	25.9	34.9	16.5	-	-	-	-	-
	✓	-	<b>40.4</b>	<b>31.7</b>	70.9	24.0	16.3	57.1	68.5	42.3	28.4	37.5	18.4	-	-	-	-	-
ABIN	✗	-	48.1	40.1	73.1	28.2	26.3	65.4	79.6	48.3	40.1	43.9	28.1	-	-	-	-	-
	✓	-	<b>49.3</b>	<b>41.2</b>	74.8	29.1	26.9	66.5	81.2	48.7	42.5	45.0	28.7	-	-	-	-	-
BetaE	✗	31.1	41.7	34.3	65.7	26.0	24.9	55.4	66.3	44.3	27.8	39.8	25.2	14.1	14.6	11.5	6.6	12.6
	✓	<b>32.6</b>	<b>43.7</b>	<b>37.0</b>	67.4	27.0	25.7	56.9	68.0	47.6	30.3	41.7	28.4	15.4	15.1	12.2	7.1	13.7
FuzzQE*	✗	31.5	41.9	33.0	67.9	27.2	25.9	56.2	67.7	46.7	30.2	34.6	20.6	15.2	15.7	12.4	7.3	13.5
	✓	<b>33.1</b>	<b>43.8</b>	<b>35.8</b>	69.2	28.2	26.3	57.9	69.4	48.0	31.5	38.5	25.2	16.4	17.1	13.5	8.0	14.3
FB15k-237																		
GQE	✗	-	16.4	10.3	35.6	7.4	5.4	23.3	34.5	16.6	10.5	8.3	5.8	-	-	-	-	-
	✓	-	<b>12.3</b>	<b>18.4</b>	38.1	9.0	6.8	25.4	36.9	19.3	12.7	10.1	7.1	-	-	-	-	-
Query2Box	✗	-	20.6	14.0	41.0	9.7	7.0	29.4	42.4	21.3	12.6	12.4	9.6	-	-	-	-	-
	✓	-	<b>22.9</b>	<b>16.3</b>	44.2	11.4	8.7	32.2	44.5	24.1	15.3	14.8	11.0	-	-	-	-	-
ABIN	✗	-	30.2	21.7	54.9	17.2	14.1	42.4	56.2	31.4	18.2	19.9	17.1	-	-	-	-	-
	✓	-	<b>32.1</b>	<b>23.2</b>	58.5	18.3	15.0	44.6	59.8	33.6	19.9	21.2	18.0	-	-	-	-	-
BetaE	✗	15.5	21	14.3	39.3	11.0	10.0	28.8	42.6	22.4	12.7	12.5	9.7	5.1	8.0	7.4	3.5	3.4
	✓	<b>17.1</b>	<b>23.1</b>	<b>16.2</b>	42.5	12.3	11.1	31.0	45.7	25.7	13.4	14.6	11.2	6.7	8.7	8.7	4.2	4.1
FuzzQE*	✗	17.1	22.8	16.6	40.7	11.8	9.8	31.4	45.5	25.0	17.4	13.9	9.9	7.3	10.2	6.8	4.9	5.2
	✓	<b>18.4</b>	<b>24.2</b>	<b>17.3</b>	43.2	13.1	10.9	33.4	47.6	25.9	18.1	14.6	10.7	8.9	11.7	7.5	5.7	6.4
NELL995																		
GQE	✗	-	18.6	12.5	33.0	12.1	9.6	27.5	35.2	18.4	14.4	8.5	8.7	-	-	-	-	-
	✓	-	<b>21.4</b>	<b>14.6</b>	39.4	13.2	10.8	31.7	38.9	20.8	17.5	10.3	9.6	-	-	-	-	-
Query2Box	✗	-	22.9	15.2	42.3	14.0	11.0	33.3	44.6	22.5	16.8	11.3	10.3	-	-	-	-	-
	✓	-	<b>25.5</b>	<b>16.5</b>	49.7	15.3	12.2	37.7	48.7	24.2	18.5	11.9	11.4	-	-	-	-	-
ABIN	✗	-	32.6	22.5	62.7	18.2	14.9	48.5	58.6	33.4	23.1	18.7	14.9	-	-	-	-	-
	✓	-	<b>34.0</b>	<b>23.8</b>	66.1	19.0	15.2	50.2	60.5	34.7	24.4	19.8	16.2	-	-	-	-	-
BetaE	✗	18.3	24.6	14.8	52.8	13.1	11.4	37.5	47.6	24.2	14.3	12.2	8.6	5.1	7.8	11.6	4.6	5.4
	✓	<b>20.2</b>	<b>27.4</b>	<b>17.3</b>	58.9	16.0	13.5	40.2	49.6	27.9	17.0	14.8	9.3	5.7	8.1	11.9	5.0	5.5
FuzzQE*	✗	20.0	27.3	18.3	55.9	16.8	14.9	36.4	48.0	26.1	19.7	15.6	11.9	7.1	8.9	10.9	3.5	4.3
	✓	<b>21.4</b>	<b>29.0</b>	<b>19.9</b>	59.4	17.2	15.4	39.2	50.2	27.7	21.4	16.8	13.6	8.7	9.5	11.2	4.0	5.1
WN18																		
GQE	✗	-	28.4	19.6	56.5	15.8	11.3	39.0	54.7	26.9	19.4	21.3	10.6	-	-	-	-	-
	✓	-	<b>30.9</b>	<b>21.1</b>	59.4	18.2	13.0	44.2	59.0	28.4	21.2	23.0	11.9	-	-	-	-	-
Query2Box	✗	-	40.1	30.8	70.2	26.2	15.3	56.6	68.9	42.4	28.9	35.4	16.7	-	-	-	-	-
	✓	-	<b>43.2</b>	<b>33.8</b>	75.6	28.6	16.9	58.8	73.2	45.8	32.2	38.1	19.2	-	-	-	-	-
ABIN	✗	-	49.8	42.6	77.1	29.6	26.7	65.2	78.9	52.4	39.7	48.6	29.7	-	-	-	-	-
	✓	-	<b>51.1</b>	<b>44.0</b>	78.4	30.2	27.5	66.9	80.2	54.4	40.6	50.2	30.8	-	-	-	-	-
BetaE	✗	32.9	42.5	32.6	70.7	27.8	25.5	57.5	70.5	43.6	30.4	37.7	18.5	18.8	17.4	15.2	10.2	16.3
	✓	<b>34.4</b>	<b>44.6</b>	<b>35.1</b>	74.8	28.4	26.2	59.2	72.8	46.3	32.8	40.0	21.1	19.1	17.5	15.5	10.8	16.5
FuzzQE*	✗	33.1	43.9	34.6	73.9	28.4	26.0	58.2	70.1	45.5	31.8	40.3	20.7	15.9	16.0	14.9	8.7	13.4
	✓	<b>34.6</b>	<b>45.6</b>	<b>36.2</b>	76.5	29.1	26.4	60.5	73.7	47.6	33.2	41.2	22.6	16.8	16.4	15.5	9.8	15.4

run, which is much more time consuming than other comparison methods. Limited by our experimental conditions and in order to be comparable, we manually reproduce the Logical Mapping Function of FuzzQE as it was described in the paper and marked as FuzzQE\* in Table 3 and Table 4.

**Implementation Details.** At the very beginning, all nodes and relationships in the knowledge graph need to be embedded onto feature space in order to participate in model training. So in order to obtain better representations of nodes and relations, we pre-train the embedding model (i.e., AcrE) and then reload parameters of embedding layer into our end-to-end process as initialization

**Table 4: Overall MRR results (%) over four datasets.**

Model	FB15k		FB15k-237		NELL995		WN18	
	avg	avg <sub>p</sub>						
GQE	-	28.0	-	16.4	-	18.6	-	28.4
Query2Box	-	37.9	-	20.6	-	22.9	-	40.1
ABIN	-	48.1	-	30.2	-	32.6	-	49.8
<i>LoD-ABIN</i>	-	<b>49.3</b>	-	<b>32.1</b>	-	<b>34.0</b>	-	<b>51.1</b>
BetaE	31.1	41.7	15.5	21.0	18.3	24.6	32.9	42.5
FuzzQE*	31.5	41.9	17.1	22.8	20.0	27.3	33.1	43.9
<i>LoD-FuzzQE*</i>	<b>33.1</b>	<b>43.8</b>	<b>18.3</b>	<b>24.2</b>	<b>21.4</b>	<b>29.0</b>	<b>34.6</b>	<b>45.6</b>

parameters. Specifically, AcrE uses  $\tau$  to denote a concatenating operation and 2-dimensional (2D) reshaping function.

$$C^0 = \omega_c^0 \star \tau([e; r]) + b_c^0 \quad (7)$$

$$C^l = \omega_c^l \star C^{l-1} + b_c^l \quad (8)$$

$$o = \text{Flatten}(\text{ReLU}(C^L + \tau([e; r]))) \quad (9)$$

where  $\star$  denotes a 2D convolution operation,  $\omega_c^0$  is a standard filter while  $\omega_c^l$  is a dilated filter,  $b_c^0$  and  $b_c^l$  are bias vectors.  $C^l$  denotes the output after  $l$  convolutions while  $C^L$  is the output of the last dilated convolution. We train the model by optimizing a listwise loss function for 20 epochs and take 512 as the embedding dimension  $d$  of nodes.

Besides, inspired by the theory of Distributional Hypothesis [35], [9, 16, 44, 60] proposes that aggregating local information to augment entity representation is helpful for KGR tasks. Therefore we search the neighbor entity set of each entity contained in reasoning sub-graphs for once, and align the maximum of neighbors as 64. Following [39], we concatenate the embedding of the center entity and neighbors and then do feature fusion by a 2D standard convolution. Suppose an entity  $e \in \mathbb{R}^d$  and its aligned neighbor set  $N_e'$ , the new representation  $e' = \text{MLP}(\text{Flatten}(\text{ReLU}(\omega_n \star (\tau'(e, N_e') + b_n)))$ , where  $\star$  denotes a 2D convolution operation,  $\omega_n$  is the filter,  $b_n$  is the bias and the specification of MLP is  $\mathbb{R}^{m_1 \times m_2} \times \mathbb{R}^d$ . We define the concatenate function  $\tau'(e, N_e) \in \mathbb{R}^{m_1 \times m_2}$  as  $[e; e_{neib}^1; e_{neib}^2; \dots; e_{neib}^m]$  where  $e_{neib}^i \in N_e'$ .

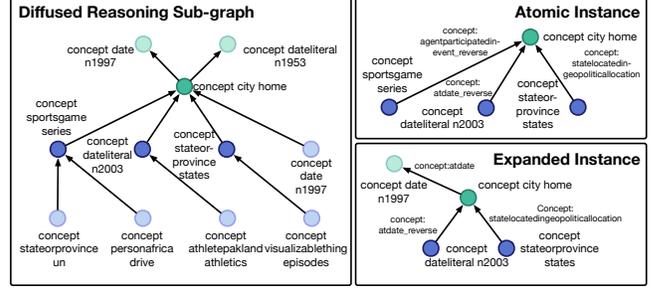
**Hyperparameters.** We adjust the following hyperparameters to obtain the best model performance. Noted that we use the same hyperparameters before and after applying *LoD* for each base model.

- Dimensions to the embeddings  $d$  from {256, 378, 512, 768, 1024}
- Learning rate lr from {1e-4, 5e-3, 1e-3}
- Batch size  $b_z$  from {128, 256, 512}
- Negative sample size  $n_s$  from {32, 64, 128}
- Maximum of neighbors  $M$  from {0, 16, 32, 64, 128}
- Length of Logic-specific prompt  $d_T$  from {8, 16, 32, 64}
- Attenuation Factor  $z$  is from {0.10, 0.05, 0.01}

The best set of *LoD* hyperparameters is shown in Table 2.

## 5.2 Performance

**Overall.** As shown in Table 3, we apply *LoD* on several mainstream models on FB15k. Table 4 shows the comparative experiments of best implementation of *LoD* on four datasets. Note that to implement *LoD* using the same backbone, we evaluate EPFO (i.e., only


**Figure 6: Illustration of a Logic Diffusion instance of 3i – ex paradigms in NELL995.**

avg<sub>p</sub>) and FOL (i.e., avg and avg<sub>p</sub>) in the top and bottom halves of Table 4 respectively. For EPFO, we implement *LoD* based on ABIN (noted as *LoD-ABIN*), while for FOL, by FuzzQE\* (noted as *LoD-FuzzQE\**). It shows that *LoD-ABIN* achieves gains of **1.2%**, **1.9%**, **1.4%** and **1.3%** in avg<sub>p</sub> MRR, *LoD-FuzzQE\** achieves **1.6%**, **1.2%**, **1.4%** and **1.5%** in avg MRR on FB15k, FB15k-237, NELL995 and WN18 respectively. Apparently, *LoD* benefits from both representation enhancement and logic augmentation by aggregating neighboring information.

**Ablation on LoD Architecture.** In this section, we perform ablation experiments of each part in LoD Architecture. To highlight the impact and to be fair, we remove the KGE pre-training and neighboring feature fusion in these experiments. As shown in Table 5, *Hier\_conj* means Hierarchical Conjunctive Query, *L\_prompt* means Logic-specific Prompt with  $d_T = 32$  and *Grad\_adapt* means Gradient Adaption with  $z = 0.05$ . The results show that *Hier\_conj*, *L\_prompt* and *Grad\_adapt* achieves gains of **0.7%**, **0.1%** and **0.2%** in avg<sub>p</sub> MRR respectively. *Hier\_conj* achieves gains of **1.6%** in avg<sub>unseen</sub> MRR. Activating all of sub-modules can achieve the final gain of **1.1%** in avg<sub>p</sub> MRR and **1.8%** in avg<sub>unseen</sub> MRR. Such results fully demonstrate the effectiveness of *LoD*. In particular, the performance of *Hier\_conj* in avg<sub>unseen</sub> MRR proves that neighbor diffusion from logic perspective has a great significant improvement on unseen FOL.

**Visualization of Logic Diffusion.** In order to show the results of logic diffusion more intuitively, here we give an example on the NELL995 dataset. As it is illustrated in Figure 6, the atomic training data as input belongs to 3i paradigm. By Hierarchical Conjunctive Query, the reasoning sub-graph is diffused as the left. Then a new training instance which belongs to unseen ip paradigm is sampled and participate in following training process by random walking. Combining with results in Table 4, additional training data pre-built by the same way contributes to generalization to unseen paradigms indeed.

## 5.3 Ablation Study with Noisy Data

**Overall.** As shown in Table 6, we evaluate our *LoD Loss* on noisy knowledge graph which consists of 80% FB15k as data of rigorous logic and 20% NELL995 as that of non-rigorous logic. Note that GQE uses a max-margin loss rather than a contrast loss where a cosine similarity is used as the similarity metric. In retrospect, we

**Table 5: Ablation MRR results (%) on FB15k. Model implementation based on Query2box.  $avg_p$  denotes the average MRR on EPFO.  $avg_{unseen}$  denotes the the average MRR on  $\{pi, ip, 2u, up\}$ .**

Hier_conj	L_prompt	Grad_adapt	$avg_p$	$avg_{unseen}$	1p	2p	3p	2i	3i	pi	ip	2u	up
			37.9	29.3	67.2	21.7	14.3	54.9	66.3	39.7	25.9	34.9	16.5
✓			38.6	30.9	67.1	21.5	14.2	54.8	66.5	41.5	27.7	36.7	17.8
	✓		38.0	29.3	67.5	21.8	14.3	55.1	66.4	39.6	25.9	35.1	16.6
		✓	38.1	29.4	67.1	21.6	14.2	55.3	66.6	39.8	26.1	35.2	16.6
	✓	✓	38.2	29.5	67.4	21.8	14.3	55.4	66.7	39.8	26.1	35.3	16.7
✓		✓	38.7	31.0	67.0	21.5	14.2	55.0	66.5	41.4	<b>28.1</b>	36.6	17.8
✓	✓		38.7	30.8	67.3	21.8	14.3	55.2	66.6	41.3	27.7	36.4	17.6
✓	✓	✓	<b>39.0</b>	<b>31.1</b>	<b>67.7</b>	<b>22.0</b>	<b>14.4</b>	<b>55.5</b>	<b>66.9</b>	<b>41.7</b>	27.8	<b>36.8</b>	<b>17.9</b>

**Table 6: MRR results (%) on noisy knowledge graph with 80% FB15k and 20% NELL995. Methods with *+dl* indicates using *LoD Loss*.  $avg_p$  indicates the the average MRR on EPFO. Decline indicates the MRR drop compared to without noise. Recovery indicates the MRR recovery by *LoD Loss*. (i.e., 7.7-2.0 = 5.7)**

Setting Up	Model	$avg_p$ Recovery $\uparrow$	$avg_p$ Decline $\downarrow$	$avg_p$ $\uparrow$	1p	2p	3p	2i	3i	pi	ip	2u	up
Setting 1	GQE	-	7.7	20.3	33.8	13.0	10.1	29.7	39.9	19.8	16.3	11.0	9.3
	GQE <i>+dl</i>	5.7	2.0	26.0	49.4	13.9	10.7	36.7	48.6	25.4	17.9	20.3	10.7
	Query2Box	-	12.1	25.9	45.7	15.8	12.2	37.5	49.8	23.4	18.2	18.7	11.6
	Query2Box <i>+dl</i>	9.8	2.3	35.7	64.6	20.4	13.6	51.4	64.5	37.2	23.6	30.7	14.9
Setting 2	GQE	-	11.8	16.2	30.4	10.7	8.4	21.3	30.5	16.1	12.5	7.9	8.0
	GQE <i>+dl</i>	6.7	5.1	22.9	51.2	14.4	10.5	36.4	18.3	25.8	18.1	20.5	10.8
	Query2Box	-	14.4	23.5	40.6	13.4	11.6	34.8	46.2	22.1	16.4	16.5	10.2
	Query2Box <i>+dl</i>	11.4	3.0	34.9	64.4	19.9	13.2	51.3	62.4	36.4	22.3	29.5	14.7

**Table 7: MRR results (%) on noisy knowledge graph. Methods with *+dl* denotes those trained with *LoD Loss*. Here  $avg_p$  denotes the the average MRR on EPFO.**

$\rho$	Model	Setting 1			Setting 2		
		Rec. $\uparrow$	Dec. $\downarrow$	$avg_p$ $\uparrow$	Rec. $\uparrow$	Dec. $\downarrow$	$avg_p$ $\uparrow$
10	GQE	-	4.5	23.5	-	8.9	19.1
	GQE <i>+dl</i>	2.9	1.6	26.4	5.0	3.9	24.1
	Query2Box	-	5.3	32.6	-	12.5	25.4
	Query2Box <i>+dl</i>	4.1	1.2	36.7	9.9	2.6	35.3
20	GQE	-	7.7	20.3	-	11.8	16.2
	GQE <i>+dl</i>	5.6	2.0	26.0	6.7	5.1	22.9
	Query2Box	-	12.1	25.9	-	14.4	23.5
	Query2Box <i>+dl</i>	9.8	2.3	35.7	11.4	3.0	34.9
30	GQE	-	10.7	17.3	-	13.8	14.2
	GQE <i>+dl</i>	6.9	3.8	24.2	7.1	6.7	21.3
	Query2Box	-	16.3	21.6	-	17.8	20.1
	Query2Box <i>+dl</i>	12.9	3.4	34.5	13.2	4.6	33.3
40	GQE	-	12.5	15.5	-	15.7	12.3
	GQE <i>+dl</i>	4.9	7.6	20.4	8.3	7.4	20.6
	Query2Box	-	18.3	19.6	-	19.5	18.4
	Query2Box <i>+dl</i>	10.3	8.0	29.9	13.4	6.1	31.8
50	GQE	-	14.6	13.4	-	17.1	10.9
	GQE <i>+dl</i>	2.8	11.8	16.2	9.3	7.8	20.2
	Query2Box	-	20.6	17.3	-	22.3	15.6
	Query2Box <i>+dl</i>	7.3	13.3	24.6	13.8	8.5	29.4

divide the noisy knowledge graph into (1) training with noise, and (2) testing with noise. Therefore, we design two settings:

- Setting 1: Distinguishing and blocking noisy data during testing, blocking delayed 50,000 steps during training. In this setting, noisy data both in train and test set.
- Setting 2: Distinguishing and blocking noisy data during testing but no sieves are used during training. In this setting, noisy data only in test set.

Since GQE and Query2Box can not handle the negation operation, we only calculate results over EPFO. As shown in Table 6, *+dl* achieves a gain of 5.7%  $avg_p$  MRR in setting 1. *+dl* achieves a gain of 6.7%  $avg_p$  MRR in setting 2. These results demonstrate that *LoD Loss* guides models for robust representation and resists the data of non-rigorous logic over noisy knowledgraph during both training and testing.

**Noise Ratio.** Since we take mixed up data of FB15k and NELL995 as noisy knowledge graph, experimentation with different noisy data percentages is also necessary. The results with  $(100 - \rho)\%$  of FB15k and  $\rho\%$  of NELL995 are shown in Tables 7, where the similarity threshold of GQE is  $\zeta = 0.15$  and Query2Box is  $\zeta = -5.0$ . For setting 1, the proposed *LoD Loss* with Query2Box achieves  $avg_p$  MRR Recovery of 4.1%, 9.8%, 12.9%, 10.3% and 7.3% when 10%, 20%, 30%, 40% and 50% of noisy data are used. For setting 2, it gains 9.9%, 11.4%, 13.2%, 13.4% and 13.8%. Similar results can be observed in other base methods. We can draw three conclusions from these results: (1) *LoD Loss* is stable and effective at various non-rigorous logic ratios; (2) From setting 1, with the use of *LoD Loss*, we don't have to clean up the training set to train a usable model; (3) From setting 2, the inference process of our model is

better resist data of non-rigorous logic, which is great valuable for practical application.

## 6 CONCLUSION

In this paper, we disassemble the open-set knowledge graph into two major problems: unseen FOL and noise-rich data. To address these issues, we propose a universal module called *LoD* which achieves representation enhancement and logic augmentation. *LoD* discovers unseen queries from surroundings and achieves dynamical equilibrium between different FOL patterns. Besides, we propose a loss function named *LoD Loss* to handle noise-rich data. Extensive experiments on four public datasets demonstrate the superiority of mainstream models with the proposed *LoD* module with mainstream knowledge graph reasoning models over state-of-the-art. Experiments on open-set knowledge graph demonstrate the superiority of *LoD Loss*. Overall, we have done a pioneering work and provided a holistic solution that can be specifically used to solve common sense reasoning on open-set knowledge graph. In future work, we will extend our approach to noisy multi-modal knowledge graph which is a more realistic scene.

## REFERENCES

- [1] Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. 2020. Complex query answering with neural link predictors. *arXiv preprint arXiv:2011.03459* (2020).
- [2] Jiaxin Bai, Zihao Wang, Hongming Zhang, and Yangqiu Song. 2022. Query2Particles: Knowledge Graph Reasoning with Particle Embeddings. *arXiv preprint arXiv:2204.12847* (2022).
- [3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf>
- [4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Adv. Neural Inform. Process. Syst.*
- [5] Zongsheng Cao, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. 2021. Dual quaternion knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 6894–6902.
- [6] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. *arXiv preprint arXiv:2005.00545* (2020).
- [7] Ling Chen, Jun Cui, Xing Tang, Yuntao Qian, Yansheng Li, and Yongjun Zhang. 2022. RLPath: a knowledge graph link prediction method using reinforcement learning based attentive relation path searching and representation learning. *Applied Intelligence* 52, 4 (2022), 4715–4726.
- [8] Wenhui Chen, Wenhan Xiong, Xifeng Yan, and William Yang Wang. 2018. Variational Knowledge Graph Reasoning. In *NAACL*.
- [9] Xiaojun Chen, Ling Ding, and Yang Xiang. 2021. Neighborhood aggregation based graph attention networks for open-world knowledge graph reasoning. *Journal of Intelligent & Fuzzy Systems* (2021), 1–12.
- [10] Xuelu Chen, Ziniu Hu, and Yizhou Sun. 2022. Fuzzy Logic Based Logical Query Answering on Knowledge Graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 3939–3948.
- [11] Xiaojun Chen, Shengbin Jia, and Yang Xiang. 2020. A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications* 141 (2020), 112948.
- [12] Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan Reddy. 2021. Probabilistic entity representation model for reasoning over knowledge graphs. *Advances in Neural Information Processing Systems* 34 (2021), 23440–23451.
- [13] Tal Friedman and Guy Broeck. 2020. Symbolic querying of vector spaces: Probabilistic databases meets relational embeddings. In *Conference on Uncertainty in Artificial Intelligence*. PMLR, 1268–1277.
- [14] Dinesh Garg, Shajith Ikkal, Santosh K Srivastava, Harit Vishwakarma, Hima Karanam, and L Venkata Subramaniam. 2019. Quantum embedding of knowledge for reasoning. *Advances in Neural Information Processing Systems* 32 (2019).
- [15] Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. *arXiv preprint arXiv:1506.01094* (2015).
- [16] Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. 2017. Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. *arXiv preprint arXiv:1706.05674* (2017).
- [17] Will Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. 2018. Embedding logical queries on knowledge graphs. *Advances in neural information processing systems* 31 (2018).
- [18] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).
- [19] Jiale Han, Bo Cheng, and Xu Wang. 2020. Open domain question answering based on text enhanced knowledge graph with hyperedge infusion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. 1475–1481.
- [20] Olaf Hartig and Ralf Heese. 2007. The SPARQL query graph model for query optimization. In *European Semantic Web Conference*. Springer, 564–578.
- [21] Zijian Huang, Meng-Fen Chiang, and Wang-Chien Lee. 2022. LinE: Logical Query Reasoning over Hierarchical Knowledge Graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 615–625.
- [22] Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramon Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, et al. 2020. Leveraging abstract meaning representation for knowledge base question answering. *arXiv preprint arXiv:2012.01707* (2020).
- [23] Edward Elson Kosasih, Fabrizio Margaroli, Simone Gelli, Ajmal Aziz, Nick Wildgoose, and Alexandra Brintrup. 2022. Towards knowledge graph reasoning for supply chain risk management using graph neural networks. *International Journal of Production Research* (2022), 1–17.
- [24] Denis Krompaß, Maximilian Nickel, and Volker Tresp. 2014. Querying factorized probabilistic triple databases. In *International Semantic Web Conference*. Springer, 114–129.
- [25] Ni Lao, Tom Mitchell, and William Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the 2011 conference on empirical methods in natural language processing*. 529–539.
- [26] Shuangyin Li, Heng Wang, Rong Pan, and Mingzhi Mao. 2021. MemoryPath: A deep reinforcement learning framework for incorporating memory component into knowledge graph reasoning. *Neurocomputing* 419 (2021), 273–286.
- [27] Lihui Liu, Boxin Du, Heng Ji, Chengxiang Zhai, and Hanghang Tong. 2021. Neural-Answering Logical Queries on Knowledge Graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1087–1097.
- [28] Mojtaba Nayyeri, Gokce Muge Cil, Sahar Vahdati, Francesco Osborne, Mahfuzur Rahman, Simone Angioni, Angelo Salatino, Diego Reforgiato Recupero, Nadezhda Vassilyeva, Enrico Motta, et al. 2021. Trans4E: Link prediction on scholarly knowledge graphs. *Neurocomputing* 461 (2021), 530–542.
- [29] Brandon Reagen, Paul Whatmough, Robert Adolf, Saketh Rama, Hyunkwang Lee, Sae Kyu Lee, José Miguel Hernández-Lobato, Gu-Yeon Wei, and David Brooks. 2016. Minierva: Enabling low-power, highly-accurate deep neural network accelerators. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 267–278.
- [30] Hongyu Ren, Hanjun Dai, Bo Dai, Xinyun Chen, Michihiro Yasunaga, Haitian Sun, Dale Schuurmans, Jure Leskovec, and Denny Zhou. 2021. Lego: Latent execution-guided reasoning for multi-hop question answering on knowledge graphs. In *International Conference on Machine Learning*. PMLR, 8959–8970.
- [31] Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *arXiv preprint arXiv:2002.05969* (2020).
- [32] Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. *Advances in Neural Information Processing Systems* 33 (2020), 19716–19726.
- [33] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. 2021. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 2 (2021), 1–49.
- [34] Ali Sadeghian, Mohammadreza Armandpour, Anthony Colas, and Daisy Zhe Wang. 2021. ChronoR: rotation based temporal knowledge graph embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 6471–6479.
- [35] Magnus Sahlgren. 2008. The distributional hypothesis. *Italian Journal of Disability Studies* 20 (2008), 33–53.
- [36] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*. Springer, 593–607.
- [37] Yingchun Shan, Chenyang Bu, Xiaojian Liu, Shengwei Ji, and Lei Li. 2018. Confidence-aware negative sampling method for noisy knowledge graph embedding. In *2018 IEEE International Conference on Big Knowledge (ICBK)*. IEEE,

- 33–40.
- [38] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 3060–3067.
- [39] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 3060–3067.
- [40] Tianyang Shao, Xinyi Li, Xiang Zhao, Hao Xu, and Weidong Xiao. 2021. DSKRL: A dissimilarity-support-aware knowledge representation learning framework on noisy knowledge graph. *Neurocomputing* 461 (2021), 608–617.
- [41] Baoxu Shi and Tim Weninger. 2018. Open-world knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.
- [42] Haitian Sun, Andrew Arnold, Tania Bedrax Weiss, Fernando Pereira, and William W Cohen. 2020. Faithful embeddings for knowledge base queries. *Advances in Neural Information Processing Systems* 33 (2020), 22505–22516.
- [43] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197* (2019).
- [44] Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 222–229.
- [45] Zhenwei Tang, Shichao Pei, Xi Peng, Fuzhen Zhuang, Xiangliang Zhang, and Robert Hoehndorf. 2022. Joint Abductive and Inductive Neural Logical Reasoning. *arXiv preprint arXiv:2205.14591* (2022).
- [46] Prayag Tiwari, Hongyin Zhu, and Hari Mohan Pandey. 2021. DAPath: Distance-aware knowledge graph reasoning based on deep reinforcement learning. *Neural Networks* 135 (2021), 1–12.
- [47] Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*. 57–66.
- [48] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*. PMLR, 2071–2080.
- [49] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, Nilesh Agrawal, and Partha Talukdar. 2020. Interact: Improving convolution-based knowledge graph embeddings by increasing feature interactions. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 3009–3016.
- [50] Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic embedding of knowledge graphs with box lattice measures. *arXiv preprint arXiv:1805.06627* (2018).
- [51] Guojia Wan, Bo Du, Shirui Pan, and Jia Wu. 2020. Adaptive knowledge subgraph ensemble for robust and trustworthy knowledge graph completion. *World Wide Web* 23, 1 (2020), 471–490.
- [52] Guojia Wan, Shirui Pan, Chen Gong, Chuan Zhou, and Gholamreza Haffari. 2021. Reasoning like human: Hierarchical reinforcement learning for knowledge graph reasoning. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*. 1926–1932.
- [53] Qi Wang, Yongsheng Hao, and Jie Cao. 2020. ADRL: An attention-based deep reinforcement learning framework for knowledge graph reasoning. *Knowledge-Based Systems* 197 (2020), 105910.
- [54] Wenhao Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A reinforcement learning method for knowledge graph reasoning. *arXiv preprint arXiv:1707.06690* (2017).
- [55] Zi Ye, Yogan Jaya Kumar, Goh Ong Sing, Fengyan Song, and Junsong Wang. 2022. A Comprehensive Survey of Graph Neural Networks for Knowledge Graphs. *IEEE Access* 10 (2022), 75729–75741.
- [56] Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. 2021. Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. *arXiv preprint arXiv:2110.04330* (2021).
- [57] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. *Advances in neural information processing systems* 32 (2019).
- [58] Wen Zhang, Jiaoyan Chen, Juan Li, Zezhong Xu, Jeff Z Pan, and Huajun Chen. 2022. Knowledge Graph Reasoning with Logics and Embeddings: Survey and Perspective. *arXiv preprint arXiv:2202.07412* (2022).
- [59] Yao Zhang, Peiyao Li, Hongru Liang, Adam Jatowt, and Zhenglu Yang. 2021. Fact-Tree Reasoning for N-ary Question Answering over Knowledge Graphs. *arXiv preprint arXiv:2108.08297* (2021).
- [60] Yongqi Zhang and Quanming Yao. 2022. Knowledge graph reasoning with relational digraph. In *Proceedings of the ACM Web Conference 2022*. 912–924.
- [61] Zhanqiu Zhang, Jie Wang, Jiajun Chen, Shuiwang Ji, and Feng Wu. 2021. Cone: Cone embeddings for multi-hop reasoning over knowledge graphs. *Advances in Neural Information Processing Systems* 34 (2021), 19172–19183.
- [62] Zhehui Zhou, Can Wang, Yan Feng, and Defang Chen. 2022. JointE: Jointly utilizing 1D and 2D convolution for knowledge graph embedding. *Knowledge-Based Systems* 240 (2022), 108100.
- [63] Lei Zou, Jinghui Mo, Lei Chen, M Tamer Özsu, and Dongyan Zhao. 2011. gStore: answering SPARQL queries via subgraph matching. *Proceedings of the VLDB Endowment* 4, 8 (2011), 482–493.