Neighbors Always Help: A Relation-aware Knowledge Reasoning Model for Inductive Relation Prediction

Anonymous ACL submission

Abstract

Inductive relation prediction, an important task for knowledge graph completion, is to predict the relations between entities that are unseen at the training stage. The latest methods use pre-trained language models (PLMs) to encode the paths between the head entity and tail entity and achieve state-of-the-art prediction performance. However, these methods cannot well handle no-path situations and are also unable to learn comprehensive representations for different relations to overcome the difficulty of inductive relation prediction. To tackle this issue, we propose a novel **R**elation-**a**ware **k**nowledg**e** **r**easoning model entitled Raker which develops an adaptive reasoning information extraction method to identify relation-aware reasoning neighbors of entities in the target triple to handle no-path situations, and enables PLMs to be aware of the predicted relation by the relation-specific soft prompting. Raker is evaluated on three public datasets and achieves SOTA performance in inductive relation prediction when compared with the baseline methods. Notably, the absolute improvement of Raker is even more than 10% on the FB15k-237 inductive setting. Moreover, Raker also demonstrates its superiority in both transductive and few-shot settings. The code of Raker will be publicly available after the double-blind review process.

1 Introduction

Knowledge graphs (KGs) are heterogeneous graphs consisting of different nodes as entities and different types of edges as relations. KGs play an essential role in a wide range of applications such as recommendation systems (Zhang et al., 2021) and question-answering (Yasunaga et al., 2021; Saxena et al., 2022). However, most KGs suffer from incompleteness, making predicting missing relations between entities in KGs a popular research problem (Ji et al., 2021; Chen et al., 2023; Liang et al., 2022).

Given an incomplete knowledge graph, the general relation prediction task is to score the probability that the target triple (h, r, t) is true, where h and t denote the head and tail entities, respectively, and r refers to a certain target relation. Specifically, inductive relation prediction is to predict the relations between entities that are unseen at the training stage. Existing methods for relation prediction can be roughly divided into 4 categories, i.e., *embedding-based methods*, *rule-based methods*, *graph-based methods*, and *PLM-based methods* (Ji et al., 2021; Chen et al., 2023). Embedding-based methods, e.g., TransE (Bordes et al., 2013), RoateE (Sun et al., 2019), encode the entities and relations into a semantic space and then design a score function to measure the possibility of the target triple based on the encoded representations. These approaches achieve good performance on some knowledge graph completion (KGC) benchmarks but are limited to the transductive setting which requires all entities and relations to be seen at the training stage (Chen et al., 2022). Rule-based methods (Meilicke et al., 2018) extract logical rules from KGs to infer whether the target triple is correct. Graph-based methods (Teru et al., 2020; Mai et al., 2021) mainly use Graph Neural Networks (GNNs) to encode the graph structures of KGs for inferring relations between entities. PLM-based methods, e.g., BERTRL (Zha et al., 2022) and KRST (Su et al., 2023), feed the KG structure information and the textual embeddings of entities and relations into PLMs for target triple prediction, and achieve state-of-the-art performance in inductive relation prediction.

Particularly, the latest PLM-based methods, e.g., BERTRL and KRST, extract the paths between entities to capture the structure information and are thus highly dependent on the connectivity of KGs. However, KGs often suffer from high incompleteness and sparsity, and there could be no paths between entities. For example, about 13% of

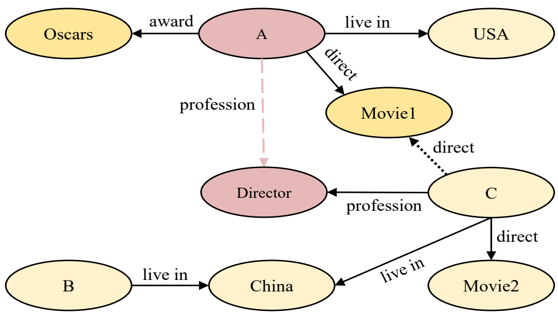

Figure 1: An example of knowledge subgraph.

entity pairs have no path connection in the widely used knowledge graph dataset FB15k-237. Without paths connecting the head entity h and the tail entity t, inferring the target triple becomes difficult. In this case, a natural idea is to add neighbor triples around entities to enrich the reasoning information for prediction. For example, as illustrated in Figure 1, we wish to predict the target triple *(A, profession, Director)*, and assume that the *direct* relation between *C* and *Movie1* is missing. By analyzing the associated triples of *A*, e.g., *(A, direct, Movie1)* and *(A, award, Oscars)*, we can infer that the target triple is most likely correct. Therefore, analyzing the relations surrounding the entities is helpful to infer the target triple. However, the contributions of neighbors associated with the head and tail entities are not equal. Some neighbors can provide strong support information for the target triple, while the others may be noisy and unreliable. For example, the neighbor triple *(A, live in, USA)* of entity *A* cannot provide strong clues for predicting *(A, profession, director)*. Therefore, identifying effective relational neighbors is an important yet challenging task.

In addition, inductive relation prediction needs rich information about the target relations because the associated entities are unseen at the training stage. PLM-based methods like BERTRL and KRST use hard prompts to directly input the relation names into PLMs, and cannot learn comprehensive representations for the target relations, which could impede PLMs' ability to be aware of the target relations in KGs.

To address the above issues, we propose the **R**elation-**a**ware **k**nowledg**e** **r**easoning model entitled Raker. Raker develops an adaptive reasoning information extraction method to adaptively extract reasoning information, i.e., reasoning paths or relation-aware reasoning neighbors, for relation prediction. In addition, Raker designs a soft prompting approach to dynamically learn comprehensive and semantic relation representations. Finally, we combine the learned relation representations and extracted reasoning information as the input sequence of PLMs for fine-tuning and relation prediction.

In sum, we make the following contributions:

- We propose a relation-aware knowledge reasoning model Raker for inductive relation prediction and adaptively extract reasoning information to address the issue of no-connection between entities in PLM-based methods.
- We propose the relation-aware reasoning neighbors extraction method to effectively identify those neighbors that are helpful for target relation prediction, and design a relation-specific soft prompting method to learn comprehensive representation for the target relation.
- We conduct extensive experiments on three public datasets. Raker outperforms the strong baseline methods by a large margin in inductive relation prediction, and also demonstrates its superiority in both transductive and few-shot settings.

2 Related Work

More details about the four categories of relation prediction methods are provided below.

Embedding-based methods. Embedding-based methods, e.g., TransE(Bordes et al., 2013), TransR(Lin et al., 2015), RoateE(Sun et al., 2019), Complex(Trouillon et al., 2016), ConvE(Dettmers et al., 2018), and TuckER(Balazevic et al., 2019), encode entities and relations as low-dimensional vectors to learn their semantic and structural information, and design certain score functions to evaluate the possibility of the target triple based on the encoded vectors. These methods are effective for transductive relation prediction (Li et al., 2023). However, they cannot generalize to unseen entities, making them unsuitable for inductive relation prediction.

Rule-based methods. Rule-based methods uncover logical rules to infer the correctness of the target triple. For example, AMIE (Galárraga et al., 2013) and RuleN (Meilicke et al., 2018), extract the inference patterns for relation prediction. NeuralLP (Yang et al., 2017) and DRUM (Sadeghian et al., 2019) employ an end-to-end approach to learn differentiable rules, and use the rules with high weights to predict the target triple.

Graph-based methods. Graph-based methods (Das et al., 2018; Schlichtkrull et al., 2018; Li et al., 2022) exploit the structure information of knowledge graphs to infer the relations between entities. For example, GraIL (Teru et al., 2020) and CoMPILE (Mai et al., 2021) extract the subgraph that encompasses the target triple and leverages GNN message passing to achieve relation prediction. DeepPath (Xiong et al., 2017) and MINERVA (Das et al., 2018) identify the paths that connect the head and tail entities of the target triple and use them to predict the missing relation. However, according to (Zhang et al., 2022), the aggregation mechanisms in GNNs are not effective for KGs.

PLM-based methods. The pre-trained language models (PLMs) like BERT (Devlin et al., 2018), T5 (Raffel et al., 2020), and GPT-3 (Brown et al., 2020) have revolutionized natural language processing, and are widely used for knowledge graph completion. For example, KG-BERT (Yao et al., 2019) fine-tunes BERT with the descriptions of entities and relations to predict the missing relations. PKGC (Lv et al., 2022) uses PLMs to encode the definition and attributes of head and tail entities for predicting the target triple. BERTRL (Zha et al., 2022) employs BERT to encode reasoning paths between head and tail entities to predict the target triple. KRST (Su et al., 2023) further introduces path extraction metrics, i.e., relation path coverage and confidence, to select relevant paths for the target triple. Specifically, BERTRL and KRST capture both structural and semantic information in knowledge graphs and achieve SOTA performance for inductive relation prediction. However, they still struggle to infer the target triple (h, r, t) when there are no paths between entities h and t, which significantly hinders their performance in real application scenarios.

3 Methodology

Figure 2 illustrates the framework of Raker which proposes the adaptive reasoning information extraction method to address no-path issue and the relation-specific soft prompting method to make PLMs aware of the predicted relations' semantic representation. Concretely, given the target triple (h, r, t), Raker primarily extracts the paths between entities h and t to obtain reasoning information. If there are no paths between the two entities, Raker extracts relation-aware reasoning neighbors as the reasoning information. Meanwhile, Raker designs a soft prompt to learn comprehensive representations for the target relation, thereby guiding the PLMs to focus on the relevant information for relation inference. Finally, Raker fine-tunes PLMs for relation prediction with the learned relation representations and extracted reasoning information.

3.1 Adaptive Reasoning Information Extraction

Given the target triple (h, r, t), Raker first tries to extract the paths between entities h and t since these paths provide effective reasoning information to evaluate the relation r (Zha et al., 2022; Su et al., 2023). If the two entities are disconnected, Raker develops the relation-aware reasoning neighbors extraction method to accurately identify those neighbor triples that are helpful for the relation prediction. In this way, we can achieve adaptive reasoning information extraction. Note that, the reasoning paths and relation-aware reasoning neighbors usually contain duplicate reasoning information. For example, to predict triple *(A, profession, Director)* in Figure 1, the reasoning path $A \xrightarrow{direct} Movie1 \xleftarrow{direct} C \xrightarrow{profession} Director$ between A and $Director$ covers the reasoning neighbor $A \xrightarrow{direct} Moive1$. Thus, Raker only uses the reasoning paths if head and tail entities are connected to reduce redundancy.

For easy representation, we denote a KG as $G{=}(E, R, D)$, where E and R represent the sets of entities and relations, respectively, and $D{=}\{(h, r, t)|h, t \in E, r{\in}R\}$ represents the triples in G.

3.1.1 Reasoning Paths Extraction

Reasoning paths can be formulated as the probability logic rules for knowledge reasoning. For example, we can easily infer *(C, mother of, B)* from rule $(A, \textit{father of}, B) \wedge (C, \textit{married}, A) \rightarrow (C, \textit{mother of}, B)$. Therefore, the paths between the head and tail entities show great reasoning power for inductive relation prediction. Following BERTRL (Zha et al., 2022) and KRST (Su et al., 2023), Raker uses the Breadth-First Search algorithm to extract the reasoning paths between head and tail entities.

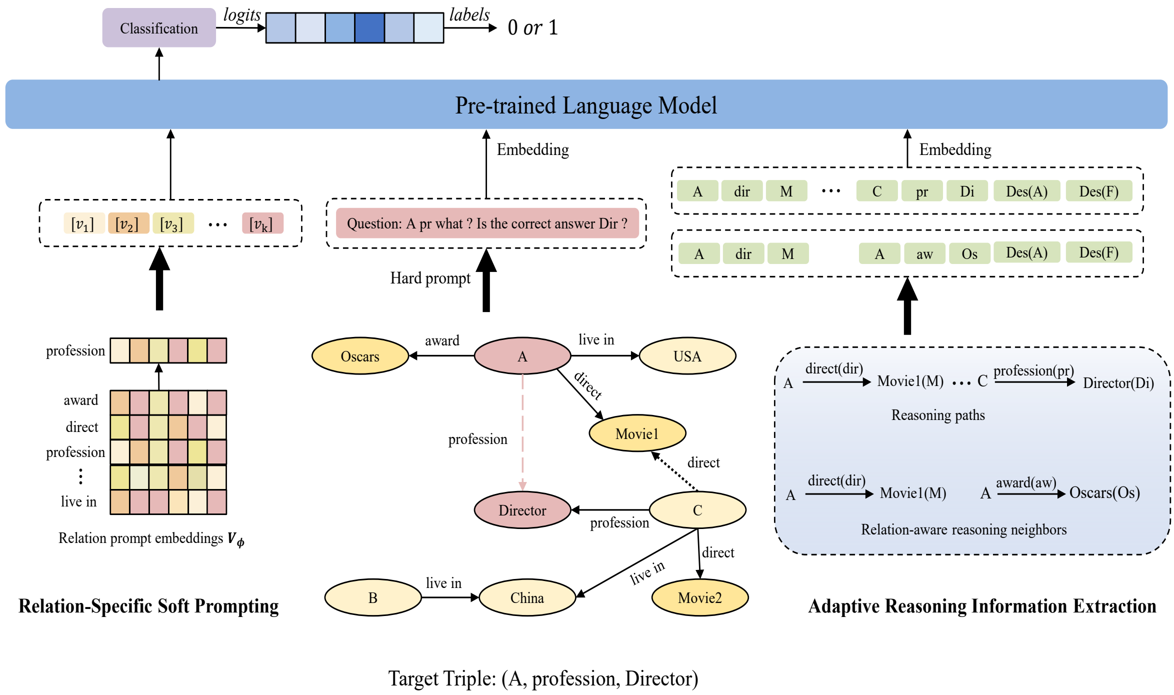

Figure 2: The framework of Raker model.

3.1.2 Relation-aware Reasoning Neighbors Extraction

To predict a target triple, the contributions of neighbors associated with the head and tail entities are not equal. As discussed previously, some neighbors can provide strong support information for the target triple, while others may be noisy and unreliable. Therefore, we propose a relation-aware reasoning neighbors extraction method to identify those neighbors associated with the head and tail entities of the target triple that are helpful to the relation prediction.

Given the target triple (h, r, t), relation-aware reasoning neighbors refer to these triples that contain h or t, and their relations could help to predict the target relation r. To identify these relation-aware reasoning neighbors for the target triple, we first calculate the relative frequency of relations for all entities in the entire KG. Then, we calculate the contribution score of each relation associated with the head and tail of the target triple. Finally, the relation-aware reasoning neighbors are extracted based on the contribution scores.

For entity e, all the relations associated with e are denoted as R^e. For each relation $r \in R^e$, the relative frequency f_r^e is calculated as below.

$$f_r^e = \frac{n_r^e}{\sum_{r_i \in R^e} n_{r_i}^e} \tag{1}$$

where n_r^e is the frequency of relation r associated with e. For example, we have $f_{direct}^C = 1/3$ for entity C and relation *direct* in Figure 1. Then, for each entity e, we can obtain a vector $\boldsymbol{F}^e$ consisting of the relative frequencies of all the relations associated with e. The vectors for all entities form a matrix $\boldsymbol{F}$.

For the target relation r, we extract the entities that are heads of r in KG G and analyze the distributions of relations associated with these entities. Concretely, we define D_r as the set of triples containing relation r, D_r^h as the set of triples that contain relation r and have h as their head entities, and E_r^H as the set of head entities appearing in D_r. In addition, we define R_r^H as the set of relations associated with entities in E_r^H, i.e., $R_r^H = \bigcup_{h_i \in E_r^H} R^{h_i}$.

For target relation r, we calculate the contribution score $p(r \mid r_i)$ of each relation $r_i \in R_r^H$ to r. Intuitively, given relation r_i, $p(r \mid r_i)$ means the probability that the head entity of r_i also has the relation r. The contribution score $p(r \mid r_i)$ is calculated as below.

$$p(r \mid r_i) = \frac{p(r)\, p(r_i \mid r)}{\sum_{r_k \in R} p(r_k)\, p(r_i \mid r_k)} \tag{2}$$

$$p(r) = \frac{|D_r|}{\sum_{r_i \in R} |D_{r_i}|} \tag{3}$$

where $p(r)$ is the appearance probability of relation r in KG G, and $|D_r|$ denotes the number of triples in D_r. The item $p\left(r_i \mid r\right)$ in Eq. (3) is calculated as below.

$$p\left(r_i \mid r\right)=\sum_{h_i \in E_r^H} s_{(r, r_i)}^{h_i} \tag{4}$$

where $s_{(r,r_i)}^{h}=f_{r_i}^{h} \times f_{inv}$, $f_{r_i}^{h}$ is the relative frequency of relation r_i associated with entity $h \in E_r^H$, and f_{inv} is the inverse frequency, i.e., the logarithm of the ratio between the total number of entities and the number of entities having relation r_i.

For example, in Figure 1, only triple (*C*, *profession*, *Director*) contains target relation *profession*. Therefore, we have $E_{pro}^{H}=\{C\}$ and $R_{pro}^{H}=$ {*profession, direct, live in*}, and just need to analyze the relations associated with *C*. Specifically, for relation *direct*, we have $f_{dir}^{C}=1/3$ and $f_{inv}=log(8/2)$ since the example KG has 8 entities and 2 entities have relation *direct* if the relation *direct* between entities *C* and *Movie1* is ignored. Finally, we have $p\left(pro \mid dir\right)=s_{(pro,dir)}^{C}=1/3 \times log(8/2)$.

After calculating the contribution score $p(r \mid r_i)$ for each relation $r_i \in R_r^H$, we use a threshold α to filter out those relations in R_r^H with contribution scores lower than α to get the reliable relations $R_r^{H'}$ associated with head entities for relation r. In addition, for target relation r, we also calculate the reliable relations, i.e., $R_r^{T'}$, associated with tail entities in the same way.

Then, we further calculate the reliable relations $R^{h'}$ and $R^{t'}$ for the head entity h and tail entity t, respectively, in the target triple, i.e.,

$$R^{h'}=R^{h} \cap R_r^{H'},\ R^{t'}=R^{t} \cap R_r^{T'} \tag{5}$$

where R^h and R^t are the sets of relations associated with entities h and t, respectively.

Finally, if both $R^{h'}$ and $R^{t'}$ are not empty, we extract the relation-aware reasoning neighbors RN for the target triple (h, r, t) as below.

$$RN=\left(\cup_{r_h \in R^{h'}} D_{r_h}^{h}\right) \cup \left(\cup_{r_t \in R^{t'}} D_{r_t}^{t}\right) \tag{6}$$

For the implementation details of relation-aware reasoning neighbors extraction, please refer to appendix A.

3.2 Relation-specific Soft Prompting

Existing PLM-based relation prediction methods usually use hard prompts to encode the target triple, e.g., "Question: **[head entity] [relation]** what ? Is the correct answer **[tail entity]** ?". Such hard prompts struggle to be aware of the predicted relation and cannot adapt to diverse triples. Furthermore, they are limited to the pre-defined set of instructions and impede the PLM's ability to leverage the internal knowledge to generalize to unseen entities in inductive relation prediction. To overcome these limitations, we propose the relation-specific soft prompting method.

Concretely, given a pre-trained language model LM_θ parametrized by θ, the input sentence embedding generated by Raker is $\boldsymbol{z}=[\boldsymbol{e}([\mathrm{CLS}])\ \boldsymbol{v}\ \boldsymbol{e}(q)\ \boldsymbol{e}([\mathrm{SEP}])\ \boldsymbol{e}(c)]$ which fuses the embedding of hard prompt q for the target triple, the embedding of adaptive reasoning information c, and the relation-specific soft prompt $\boldsymbol{v}$, where $\boldsymbol{e}(\cdot)$ denoted as embedding operation.

During the training, the i^{th} input $\boldsymbol{l}_i^j$ for the j^{th} layer of the PLM in Raker is calculated as below.

$$\boldsymbol{l}_i^j=\begin{cases} \boldsymbol{v}_i^r & (0 \leq i<k) \wedge (j=0) \\ \boldsymbol{z}_i & (i \geq k) \wedge (j=0) \\ \mathrm{LM}_\theta\left(\boldsymbol{h}_{:}^{j-1}\right)_i & \text{Otherwise} \end{cases} \tag{7}$$

where $j=0$ corresponds to the input layer, $\mathrm{LM}_\theta(\cdot)$ is the forward function of language model layer, $\boldsymbol{v}_i^r$ denotes the i^{th} relation-specific soft prompt vector, k is the number of $\boldsymbol{v}^r$. Given the target triple(h, r, t), $\boldsymbol{v}^r$ is generated based on the matrix $\boldsymbol{V_\phi}$ for relations, i.e.,

$$\boldsymbol{v^r}=U_r(\boldsymbol{V_\phi}) \tag{8}$$

where $\boldsymbol{V_\phi} \in \mathbb{R}^{|R| \times k \times m}$, $|R|$ denotes the number of distinct relations in KG G, m is the dimension of each soft prompt vector and set to 768, and $U_r(\cdot)$ denotes the transformation function for generating the specific vector for relation r.

Raker leverages the trainable matrix $\boldsymbol{V_\phi}$ to dynamically learn relation-specific representations which provide more targeted and reliable contextual information for the target triple. By combining the soft prompts with hard prompts, Raker can enhance PLMs' awareness of relation for better relation prediction.

3.3 Triple Prediction via PLM

3.3.1 Input Sentence Formation

Raker combines prompts and adaptive reasoning information to generate the input sentences for PLMs, i.e.,

$$\boldsymbol{z}=[\boldsymbol{e}([\mathrm{CLS}])\ \boldsymbol{v}\ \boldsymbol{e}(q)\ \boldsymbol{e}([\mathrm{SEP}])\ \boldsymbol{e}(c)]$$

where $\boldsymbol{v}$ is the relation-specific soft prompt, q is the hard prompt for the target triple, and c is the adaptive reasoning information. Adaptive reasoning information c can be c_p or c_n, where c_p represents the reasoning paths and c_n represents the relation-aware reasoning neighbors.

An input sentence embedding example of $\boldsymbol{z}$ w.r.t. the target triple *(A, profession, Director)* in Figure 1 is shown as follows.

$$\begin{aligned}
\boldsymbol{z} &= [\boldsymbol{e}([\text{CLS}])\ \boldsymbol{v}^{pro}\ \boldsymbol{e}(q)\ \boldsymbol{e}([\text{SEP}])\ \boldsymbol{e}(c)] \\
q &= \text{Question: A profession what? Is the correct answer} \\
&\quad \text{Director ?} \\
c &= [c_p \ or \ c_n] \\
c_p &= \text{A direct Movie1; C direct Movie1; C profession} \\
&\quad \text{Director [SEP] Des(A) [SEP] Des(Director)} \\
c_n &= \text{A direct Movie1; A award Oscars [SEP] Des}(A) \\
&\quad \text{[SEP] Des}(\mathit{Director})
\end{aligned} \quad (418)$$

where $\text{Des}(\cdot)$ is the textual description of the entity that is used to augment the reasoning information for the target triple (Yao et al., 2019; Wang et al., 2022). If the relation *direct* between entities *C* and *Movie1* exits, we have $c = c_p$, otherwise $c = c_n$.

3.3.2 Relation Prediction

After input sentence formation, Raker uses one linear layer on top of [CLS] to score the target triple's correctness, which can be regarded as a binary classification task. If reasoning paths are used, following BERTRL and KRST, Raker takes each reasoning path as a separate input to the PLM. We define $\boldsymbol{z}(c_{p_i})$ as the input sentence embedding using the i^{th} reasoning path, and the corresponding prediction probability is pro_i, i.e.,

$$pro_i = pro(y|\boldsymbol{z}(c_{p_i})),\ i = 1, 2, ..., N \quad (9)$$

where $y \in \{0, 1\}$ is class label, and N is the number of reasoning paths. Then, the final score of target triple (h, r, t) is calculated as below.

$$\text{score}(h, r, t) = \max_{i=1,2,...,N} pro(y = 1 \mid \boldsymbol{z}(c_{p_i})) \quad (10)$$

If relation-aware reasoning neighbors are used as the reasoning information, we have $\text{score}(h, r, t) = pro(y|\boldsymbol{z}(c_n))$.

We follow the negative sampling strategy in BERTRL to produce negative samples, i.e., randomly sampling entities from the common 3-hop entities of head and tail entities to corrupt the head or tail of each positive triple.

Raker is trained based on the cross entropy loss, i.e.,

$$\mathcal{L} = -\sum_{\tau} (y_\tau \log p + (1 - y_\tau) \log (1 - p)) \quad (11)$$

where $y_\tau \in \{0, 1\}$ indicates the negative or positive label.

4 Experiments

4.1 Datasets

We conducted extensive experiments on three widely used knowledge graph completion datasets, i.e., FB15k-237 (Toutanova et al., 2015), WN18RR (Dettmers et al., 2018), and NELL-995 (Xiong et al., 2017). Table 1 presents the details of three datasets. We use the inductive, transductive, few-shot subsets of these three datasets according to the setting in BERTRL (Zha et al., 2022).

Table 1: Statistics of three datasets.

Dataset	*KG*	Relations	Entities	Triples	Avg. degree
WN18RR	train	9	2746	6670	4.86
	train-2000	9	1970	2002	2.03
	train-1000	9	1362	1001	1.47
	test-transductive	7	962	638	1.32
	test-inductive	8	922	1991	4.32
FB15k-237	train	180	1594	5223	6.56
	train-2000	180	1280	2008	3.14
	train-1000	180	923	1027	2.23
	train-rel50	50	1310	3283	5.01
	train-rel100	100	1499	3895	5.20
	test-transductive	102	550	492	1.79
	test-inductive	142	1093	2404	4.40
NELL-995	train	88	2564	10063	7.85
	train-2000	88	1346	2011	2.99
	train-1000	88	893	1020	2.28
	test-transductive	60	1936	968	1.00
	test-inductive	79	2086	6621	6.35

4.2 Experiment Settings

Raker is implemented based on the Bert-uncased-base using PyTorch, and trained on two NVIDIA GeForce RTX 3090 GPUs. Following the evaluation in Grail (Teru et al., 2020) and BERTRL(Zha et al., 2022), we measure the Mean Reciprocal Rank (MRR) and Hits@1 of one positive triple among 50 samples with 49 negative triples. MRR calculates the average reciprocal rank of all positive triples and Hits@1 calculates the percentage of cases where the positive triple appears as the top-1 ranked triple. Following BERTRL, we randomly generate negative triples and use them for training and validation. For a fair comparison, we directly use the negative triples provided by BERTRL for

Table 2: Results of transductive and inductive relation prediction.

		Transductive			Inductive		
		WN18RR	FB15k-237	NELL-995	WN18RR	FB15k-237	NELL-995
	RuleN	0.669	0.674	0.736	0.780	0.462	0.710
	GRAIL	0.676	0.597	0.727	0.799	0.469	0.675
	MINERVA	0.656	0.572	0.592	-	-	-
	TuckER	0.646	0.682	0.800	-	-	-
MRR	KG-BERT	-	-	-	0.547	0.500	0.419
	BERTRL	0.683	0.695	0.781	0.792	0.605	0.808
	KRST	0.899	0.720	0.800	0.890	0.716	0.769
	Raker	**0.912**	**0.784**	**0.813**	**0.930**	**0.817**	**0.835**
	Absolute Imp.	**1.3%↑**	**6.4%↑**	**1.3%↑**	**4.0%↑**	**10.1%↑**	**2.7%↑**
	RulN	0.646	0.603	0.636	0.745	0.415	0.638
	GRAIL	0.644	0.494	0.615	0.769	0.390	0.554
	MINERVA	0.632	0.534	0.553	-	-	-
	TuckER	0.600	0.615	0.729	-	-	-
Hit@1	KG-BERT	-	-	-	0.436	0.341	0.244
	BERTRL	0.655	0.620	0.686	0.755	0.541	0.715
	KRST	0.835	0.639	0.694	0.809	0.600	0.649
	Raker	**0.853**	**0.701**	**0.730**	**0.888**	**0.729**	**0.748**
	Absolute Imp.	**1.8%↑**	**6.2%↑**	**3.6%↑**	**7.9%↑**	**12.9%↑**	**3.3%↑**

testing. Each experiment is run twice and the mean results are reported. We set the learning rate to 5×10^{-5}, reliable neighbors threshold $\alpha = 0.5$, and the length of relation-specific soft prompt $k = 10$.

4.3 Results of Transductive and Inductive Relation Prediction

Table 2 presents the results of both transductive and inductive relation prediction. Since WN18RR dataset has only 9 relations, and the paths between entities could be highly redundant, we employ the path filtering strategy in KRST model to reduce such redundancy.

According to the results in Table 2, Raker achieves the best performance among all methods, and largely outperforms the baselines. Especially, the improvement of Raker is more than 10% on the FB15k-237 inductive subset which has the largest number of distinct relations among three datasets, and has many entity pairs that are not connected. The relation-specific soft prompting and relation-aware-reasoning neighbors together contribute to such improvement.

4.4 Results of Few-shot Relation Prediction

For few-shot relation prediction, Raker follows BERTRL to extract reasoning paths on the subgraphs and Raker* follows KRST to extract reasoning paths on the entire KG graph. According to the results in Table 3, Raker and Raker* outperform most baseline methods over three datasets. In general, Raker* performs better than Raker because extracting reasoning paths from the entire KG graph could learn more information about the target triple. KRST performs best on the NELL-995 dataset for transductive relation prediction since most entity pairs in this dataset are connected and the contribution from relation-aware reasoning neighbors is thus limited.

4.5 Unseen Relation Prediction

Raker leverages a pre-trained language model for relation prediction, and has the potential to predict unseen relations. Table 4 presents the results of unseen relation prediction on the subsets of FB15k-237 with a zero-shot setting introduced by BERTRL. Following BERTRL, we use the triples with 50 types of relations and 100 types of relations for training, and testing on the FB15k-237 inductive dataset. According to the results, Raker largely outperforms BERTRL and KRST since it can extract more reasoning information for relation prediction.

4.6 Ablation Study

Table 5 shows the results of Raker after removing the relation-aware reasoning neighbors extraction method and the relation-specific soft prompting method. Obviously, after removing either of the two components, the performance of Raker de-

Table 3: Results of few-shot relation prediction.

		Transductive						Inductive					
		WN18RR		FB15k-237		NELL-995		WN18RR		FB15k-237		NELL-995	
		1000	2000	1000	2000	1000	2000	1000	2000	1000	2000	1000	2000
MRR	RuleN	0.567	0.625	0.434	0.577	0.453	0.609	0.681	0.773	0.236	0.383	0.334	0.495
	GRAIL	0.588	0.673	0.375	0.453	0.292	0.436	0.652	0.799	0.380	0.432	0.458	0.462
	MINERVA	0.125	0.268	0.198	0.364	0.182	0.322	-	-	-	-	-	-
	TuckER	0.258	0.448	0.457	0.601	0.436	0.577	-	-	-	-	-	-
	KG-BERT	-	-	-	-	-	-	0.471	0.525	0.431	0.460	0.406	0.406
	BERTRL	0.662	0.673	0.618	0.667	0.648	0.693	0.765	0.777	0.526	0.565	0.736	0.744
	KRST	0.871	0.882	0.696	0.701	**0.743**	**0.781**	0.886	0.878	0.679	0.680	0.745	0.738
	Raker	0.810	0.850	0.670	0.728	0.673	0.757	**0.892**	**0.917**	0.637	0.687	**0.750**	**0.783**
	Raker*	**0.877**	**0.887**	**0.731**	**0.736**	0.718	0.751	0.891	0.910	**0.701**	**0.723**	0.662	0.727
Hit@1	RuleN	0.548	0.605	0.374	0.508	0.365	0.501	0.649	0.737	0.207	0.344	0.282	0.418
	GRAIL	0.489	0.633	0.267	0.352	0.198	0.342	0.516	0.769	0.273	0.351	0.295	0.298
	MINERVA	0.106	0.248	0.170	0.324	0.152	0.284	-	-	-	-	-	-
	TuckER	0.320	0.415	0.407	0.529	0.392	0.520	-	-	-	-	-	-
	KG-BERT	-	-	-	-	-	-	0.364	0.404	0.288	0.317	0.236	0.236
	BERTRL	0.621	0.637	0.517	0.583	0.526	0.582	0.713	0.731	0.441	0.493	0.622	0.628
	KRST	0.790	0.810	0.611	0.602	**0.628**	**0.678**	0.811	0.793	0.537	0.524	0.637	0.629
	Raker	0.745	0.783	0.590	0.629	0.545	0.657	**0.835**	**0.864**	0.531	0.578	**0.641**	**0.683**
	Raker*	**0.815**	**0.823**	**0.621**	**0.632**	0.589	0.637	0.819	0.850	**0.566**	**0.593**	0.505	0.598

Table 4: Results of unseen relation prediction.

	Method	50 relations	100 relations
MRR	KG-BERT	-	-
	BERTRL	0.580	0.612
	KRST	0.660	0.692
	Raker	**0.714**	**0.769**
Hit@1	KG-BERT	0.266	0.450
	BERTRL	0.534	0.585
	KRST	0.551	0.560
	Raker	**0.619**	**0.668**

creases dramatically, which indicates the effectiveness and necessity of the two components.

Table 5: Results of ablation studies on FB15k-237-inductive dataset.

Method	MRR	Hit@1
w/o Relation-aware reasoning neighbors	0.720	0.643
w/o Relation-specific soft prompts	0.746	0.646
Raker	**0.817**	**0.729**

5 Conclusion

In this work, we propose the relation-aware knowledge reasoning model Raker for inductive relation prediction, and adaptively extract reasoning information to address the issue of no-connection between entities. Raker introduces the relation-aware reasoning neighbors extraction method to effectively identify those neighbors that are helpful for target relation prediction, and designs a relation-specific soft prompting method to learn comprehensive representation for the target relation. According to the experiment results under different settings, Raker largely outperforms the baseline methods in both inductive relation prediction and transductive relation prediction, and also achieves good performance for few-shot setting and unseen relation prediction.

Limitations

Although Raker can well address the issue of no-path between entities and largely outperforms baseline methods, it still has two limitations. First, Raker extracts paths and neighbors for each triple, and could be of high computational complexity if applied to predict missing entities. Second, reasoning paths and neighbors are used alternatively in Raker, and better integration methods are expected to take their advantage while avoiding redundancy.

References

Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5184–5193. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

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 *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Mingyang Chen, Wen Zhang, Yuxia Geng, Zezhong Xu, Jeff Z. Pan, and Huajun Chen. 2023. Generalizing to unseen elements: A survey on knowledge extrapolation for knowledge graphs. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 6574–6582. ijcai.org.

Mingyang Chen, Wen Zhang, Yushan Zhu, Hongting Zhou, Zonggang Yuan, Changliang Xu, and Huajun Chen. 2022. Meta-knowledge transfer for inductive knowledge graph embedding. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 927–937. ACM.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1811–1818. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 413–422. International World Wide Web Conferences Steering Committee / ACM.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514.

Ren Li, Yanan Cao, Qiannan Zhu, Guanqun Bi, Fang Fang, Yi Liu, and Qian Li. 2022. How does knowledge graph embedding extrapolate to unseen data: A semantic evidence view. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 5781–5791. AAAI Press.

Rui Li, Xu Chen, Chaozhuo Li, Yanming Shen, Jianan Zhao, Yujing Wang, Weihao Han, Hao Sun, Weiwei Deng, Qi Zhang, and Xing Xie. 2023. To copy rather than memorize: A vertical learning paradigm for knowledge graph completion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 6335–6347. Association for Computational Linguistics.

Ke Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, Xinwang Liu, and Fuchun Sun. 2022. Reasoning over different types of knowledge graphs: Static, temporal and multi-modal. *arXiv preprint arXiv:2212.05767*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2181–2187. AAAI Press.

Xin Lv, Yankai Lin, Yixin Cao, Lei Hou, Juanzi Li, Zhiyuan Liu, Peng Li, and Jie Zhou. 2022. Do pre-trained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3570–3581.

Sijie Mai, Shuangjia Zheng, Yuedong Yang, and Haifeng Hu. 2021. Communicative message passing for inductive relation reasoning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4294–4302. AAAI Press.

Christian Meilicke, Manuel Fink, Yanjie Wang, Daniel Ruffinelli, Rainer Gemulla, and Heiner Stuckenschmidt. 2018. Fine-grained evaluation of rule- and embedding-based systems for knowledge graph completion. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I*, volume 11136 of *Lecture Notes in Computer Science*, pages 3–20. Springer.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. 2019. DRUM: end-to-end differentiable rule mining on knowledge graphs. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15321–15331.

Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. 2022. Sequence-to-sequence knowledge graph completion and question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2814–2828. Association for Computational Linguistics.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Zhixiang Su, Di Wang, Chunyan Miao, and Lizhen Cui. 2023. Multi-aspect explainable inductive relation prediction by sentence transformer. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 6533–6540. AAAI Press.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Komal K. Teru, Etienne G. Denis, and William L. Hamilton. 2020. Inductive relation prediction by subgraph reasoning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 9448–9457. PMLR.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1499–1509. The Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022. Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 4281–4294. Association for Computational Linguistics.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 564–573. Association for Computational Linguistics.

Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2319–2328.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for Knowledge Graph Completion. ArXiv:1909.03193 [cs].

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 535–546. Association for Computational Linguistics.

Hanwen Zha, Zhiyu Chen, and Xifeng Yan. 2022. Inductive relation prediction by BERT. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 5923–5931. AAAI Press.

Ningyu Zhang, Qianghuai Jia, Shumin Deng, Xiang Chen, Hongbin Ye, Hui Chen, Huaixiao Tou, Gang Huang, Zhao Wang, Nengwei Hua, and Huajun Chen. 2021. Alicg: Fine-grained and evolvable conceptual graph construction for semantic search at alibaba. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 3895–3905. ACM.

Zhanqiu Zhang, Jie Wang, Jieping Ye, and Feng Wu. 2022. Rethinking graph convolutional networks in knowledge graph completion. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 798–807. ACM.

A Implementation of Relation-aware Reasoning Neighbors Extraction

Algorithm 1 provides the implementation details of the relation-aware reasoning neighbors extraction method.

Algorithm 1 Relation-aware Reasoning Neighbors Extraction

Input: KG G, target triple (h, r, t), and α
Output: Relation-aware reasoning neighbors

1: Get R_r^H and E_r^H, initialize $R_r^{H'}$ and $R_r^{T'}$ as list()
2: Calculate $\boldsymbol{F}$ as Eq. (1)
3: **for** $h \in E_r^H$ **do**
4:     **for** $r_i \in \boldsymbol{F}[h]$ **do**
5:         // $len(\boldsymbol{F}[e][r_i])$ denotes numbers of entities has relation r_i
6:         $f_{inv} = log(len(\boldsymbol{F})/len(\boldsymbol{F}[\mathbf{e}][\boldsymbol{r_i}]))$
7:         $p\,(r_i \mid r) += f_{r_i}^h \times f_{inv}$
8:     **end for**
9: **end for**
10: **for** $r_i \in R_r^H$ **do**
11:     Calculate score $p\,(r \mid r_i)$ as Eq. (2)
12:     **if** $p\,(r \mid r_i) >= \alpha$ **then**
13:         $R_r^{H'}.append(r_i)$
14:     **end if**
15: **end for**
16: Repeat step 3-15 for calculating $R_r^{T'}$
17: Calculate $R^{h'}$ and $R^{t'}$ as Eq. (5)
18: **if** $R^{h'}$ is not empty and $R^{t'}$ is not empty **then**
19:     **return** RN as Eq. (6)
20: **else**
21:     **return** empty list()
22: **end if**