Open Relation Modeling: Learning to Define Relations between Entities

Jie Huang[1] **Kevin Chen-Chuan Chang**[1] **Jinjun Xiong**[2] **Wen-mei Hwu**[1,3]
[1]University of Illinois at Urbana-Champaign, USA
[2]University at Buffalo, USA
[3]NVIDIA, USA
{jeffhj, kcchang, w-hwu}@illinois.edu
jinjun@buffalo.edu

Abstract

Relations between entities can be represented by different instances, e.g., a sentence containing both entities or a fact in a Knowledge Graph (KG). However, these instances may not well capture the general relations between entities, may be difficult to understand by humans, even may not be found due to the incompleteness of the knowledge source. In this paper, we introduce the *Open Relation Modeling* problem–given two entities, generate a coherent sentence describing the relation between them. To solve this problem, we propose to teach machines to generate definition-like relation descriptions by letting them learn from defining entities. Specifically, we fine-tune Pre-trained Language Models (PLMs) to produce definitions conditioned on extracted entity pairs. To help PLMs reason between entities and provide additional relational knowledge to PLMs for open relation modeling, we incorporate reasoning paths in KGs and include a reasoning path selection mechanism. Experimental results show that our model can generate concise but informative relation descriptions that capture the representative characteristics of entities.[1]

1 Introduction

People are always interested in relations between entities. To learn about a new concept, people want to know how this concept relates to the ones they are familiar with; when getting two related entities of interest, people ask how exactly they are related.

However, although existing systems identify related entities, they do not provide features for exploring relations between entities. For instance, in Figure 1, the top is the *ScienceDirect Topics* feature of Elsevier, which lists several related terms without any annotation; the bottom is the "*see also*" feature of Wikipedia, where the annotation of *deep learning* is not specific to the context of *natural language processing*. Users cannot get how *deep learning* and *NLP* are related by reading the annotation, while *deep learning* is used heavily recently for *NLP*.

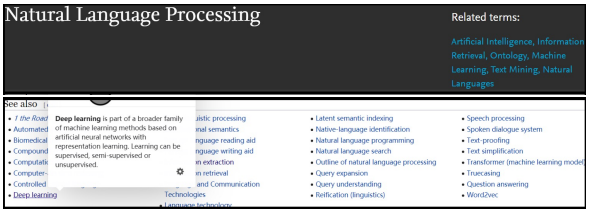

Figure 1: Examples of two current services: *Elsevier's ScienceDirect Topics* (top) and *Wikipedia's "see also"* (bottom), both of which lack open relation modeling.

Besides, even relations are represented, they may not be interpretable to humans. There are different ways to represent relations between entities. For example, if two entities co-occur in a sentence, they are possibly related and the relation can be implied by the sentence. From a structured perspective, a relation can be represented as a fact or a multi-hop reasoning path between two entities in a Knowledge Graph (KG). However, for humans without too much prior knowledge about the entities, it is still difficult to understand the relations by reading them. For example, from sentence "*we study data mining and database.*" or fact "(data mining, *facet of*, database)", humans can guess *data mining* and *database* are related fields, but they cannot know exactly how they are related. Besides, due to the limited size of the corpus or the incompleteness of the KG, for many related entities, we may not extract a sentence or a fact containing both entities.

Based on the above observation, a system for exploring relations between entities needs to meet the following requirements: 1) **interpretability**: providing interpretable relation descriptions, with which humans can easily understand relations between entities; 2) **openness**: dealing with a wide range of related entities, including those neither co-occur in a corpus nor be connected in a knowledge

[1]Code and data are available at https://github.com/jeffhj/open-relation-modeling.

Findings of the Association for Computational Linguistics: ACL 2022, pages 297 - 308

graph, where types of relations are not required to be explicitly pre-specified.

To achieve a system meeting with the above requirements, we introduce a novel task– ***Open Relation Modeling***, i.e., generating coherent sentences describing general relations between entities, where types of relations do not need to be pre-specified. Different from open relation extraction, which aims to *extract* relational facts between entities from an open-domain corpus (Banko et al., 2007), open relation modeling aims to *generate* a concise but informative sentence, capturing the representative characteristics of the given entities and their relation. From the perspective of interpretability, compared to open relation extraction whose outputs are phrases with low interpretability, e.g., *(data mining methods, to be integrate within, the framework of traditional database systems)* by Ollie (Schmitz et al., 2012), open relation modeling improves the interpretability of entity relations. For example, for *data mining* and *database*, we want to generate a sentence like "data mining *is a process of extracting and discovering patterns in large data sets involving methods at the intersection of machine learning, statistics, and* database *systems*." Such a relation description is informative and easy to understand since it contains important and precise information about entities and their relation.

To solve the task, we propose to teach machines to learn from defining entities. Definitions of entities are highly summarized sentences that capture the most representative characteristics of entities, where the general relations between the defined entity and other entities in the definitions are well captured. Therefore, we suggest to ***find the general relation between two entities by defining one entity in terms of the other entity***. To achieve this, we first collect definitions of entities and extract entity pairs from the definitions. Then we teach machines to generate definition-like relation descriptions by training a language generation model to produce definitions of entities conditioned on extracted entity pairs.

To generate informative relation descriptions, machines need knowledge about entities and relations. Therefore, we apply Pre-trained Language Models (PLMs) (Radford et al., 2019; Brown et al., 2020; Lewis et al., 2020a; Raffel et al., 2020), which have recently been shown to contain rich relational knowledge of entities (Petroni et al., 2019; Roberts et al., 2020; Wang et al., 2020; Liu et al., 2021a). To utilize knowledge to describe relations between entities, machines also need to reason between entities. We incorporate reasoning paths in KGs to help PLMs do multi-hop reasoning and provide additional relational knowledge to PLMs. We also design a reasoning path selection mechanism by confidence estimation of PLMs to select interpretable and informative reasoning paths, which are then incorporated by PLMs for open relation modeling.

We conduct both quantitative and qualitative experiments. Experimental results show that, after learning from definitions of entities, PLMs have a great ability to describe relations between entities concisely and informatively. By incorporating reasoning paths and including the reasoning path selection mechanism, machines can often generate relation descriptions well capturing relations between entities, with only minor errors that do not affect the understanding of relations. We also conduct error analysis for the proposed methods and suggest several directions for future work.

2 Open Relation Modeling

2.1 Problem Statement

The problem of ***Open Relation Modeling*** can be described as: given two entities x and y, corresponding to *head* and *tail*, the task is to generate a coherent sentence s that describes the general relation between x and y, where types of relations do not need to be pre-specified. More specifically, the expected output is a concise but informative sentence that captures the representative characteristics of the entities and their relation (examples of *data mining* and *database* as shown in Section 1).

2.2 Open Relation Modeling: Learning from Definitions

We formulate open relation modeling as a conditional sentence generation task, i.e., generating sentences capturing general relations between entities conditioned on entity pairs. Formally, we apply the standard sequence-to-sequence formulation: given an entity pair (x, y), the probability of the output relation description $s = [w_1, \ldots, w_m]$ is calculated as:

$$P(s|x, y) = \prod_{i=1}^{m} P(w_i|w_0, w_1, \ldots, w_{i-1}, x, y),$$

where w_0 is a special start token.

To generate a sentence capturing the general relation between x and y, machines need to know the semantic meanings of x and y, reason between them, and learn to describe their relation in a concise but informative form. Definitions of entities, which are highly summarized (i.e., concise but informative) sentences, capture the most representative characteristics of entities. To define an entity, other entities may be included, and the relations between the defined entity and other entities are well captured.

Therefore, we propose to *teach machines to describe relations between entities by letting them learn from defining entities*. The key idea is to *find the general relation between two entities by defining one entity in terms of the other entity*. To achieve this, we first collect definitions of entities and extract entity pairs from these definitions to form entities-definition pairs (more details are in Section 3.1). After that, we teach machines to generate relation descriptions with the desired characteristics by training a language generation model to produce definitions of entities conditioned on extracted entity pairs.

With the key idea in mind, the next step is to design the generation model. Recently, Bevilacqua et al. (2020) show that, by fine-tuning with context-gloss pairs, pre-trained language generation models can generate the glosses/definitions for definiendums that are not seen in the training data. Besides, recent studies (Petroni et al., 2019; Wang et al., 2020; Liu et al., 2021a) demonstrate that pre-trained language models contain rich relational knowledge, and such relational knowledge is essential to describing relations between entities.

Therefore, we apply pre-trained language models for open relation modeling. Particularly, we employ BART (Lewis et al., 2020a)– a recent transformer-based encoder-decoder model. In our framework, we train BART to produce the definitions of entities with extracted entity pairs as input. Specifically, we encode the entity pair (x, y) as $x; y$, e.g., $Haste; Germany$, and fine-tune the model to generate the corresponding sentence s, e.g., "*Haste is a municipality in the district of Schaumburg, in Lower Saxony, Germany*". By fine-turning on the training data, the model can learn the knowledge about entities and learn to connect two entities in a coherent sentence based on its "knowledge". When given a new entity pair, the model can generate a definition-like relation description that possesses the desired characteristics. We refer to this model as **RelationBART-Vanilla**.

2.3 Reasoning Path-Enriched Relation Modeling

While PLMs can generate coherent relation descriptions with fine-tuning on the entities-definition pairs, their ability is still limited. Recent studies (Forbes et al., 2019; Zhou et al., 2020; Richardson and Sabharwal, 2020) show that it is difficult for PLMs to reason based on their knowledge. Besides, although PLMs contain rich relational knowledge implicitly, they cannot recover all the relational knowledge in a knowledge base.

Knowledge graphs, in contrast, contain rich relational knowledge explicitly. Relations between entities can be represented by reasoning paths extracted from KGs directly. A good reasoning path can guide PLMs to do multi-hop reasoning and provide additional relational knowledge to PLMs for open relation modeling.

Therefore, we want to inject relational knowledge of KGs into PLMs and incorporate reasoning paths to help PLMs reason between entities. We achieve this by a simple encoding scheme without changing the architecture of PLMs and re-pre-training. Given a knowledge graph G, for an entity pair (x, y), if there exists a reasoning path $p(x, y) = \{x, r_1, e_1, r_2, \ldots, r_k, y\}$ in G, we encode (x, y) as $x;\ r_1\text{:}\ e_1;\ r_2\text{:}\ \ldots;\ r_k\text{:}\ y$; if not, we encode (x, y) as $x;\ \textit{unknown:}\ y$. With fine-tuning on the path-sentence pairs, the model can learn to utilize the relational knowledge in a reasoning path to reason between two entities and generate a coherent sentence describing the relation between them.

However, there may exist multiple reasoning paths between two entities in a KG, while not all reasoning paths are equally helpful. Among the reasoning paths between two entities, the shortest one usually indicates the most direct relation. For example, if two entities have a direct relation in a KG, the shortest reasoning path should be a 1-hop path $p(x, y) = \{x, r_1, y\}$. This path can represent a reasonable relation between two entities because this is the reason why the KG includes such a fact. Based on this observation, formally, given an entity pair (x, y), the selected reasoning path is

$$\hat{p}(x, y) = \underset{p(x,y) \in \mathcal{P}(x,y)}{\arg\min} \; len(p(x, y)),$$

where $\mathcal{P}(x, y)$ is the set of reasoning paths connecting x and y extracted from the KG and $len(\cdot)$

is the length of the reasoning path. We name the model trained with the shortest reasoning paths[2] as **RelationBART-SP**. To keep the presented model simple and easy to be verified, we leave the more complex mechanism of sampling reasoning paths as future work (Lao et al., 2011; Xiong et al., 2017; Chen et al., 2018). In the next section, we will show that PLMs can select interpretable and informative reasoning paths automatically based on confidence estimation.

2.4 Open Relation Modeling with Reasoning Path Selection

While shortest reasoning paths can represent the most direct relations between entities, from the perspective of human/machine understanding, these paths may not be the most interpretable and informative. For instance, given entity pair $(\textit{Haste}, \textit{Germany})$, with sentence description $s =$ "*Haste is a municipality in the district of Schaumburg, in Lower Saxony, Germany*", the shortest reasoning path in Wikidata KG is $p_1 = \{\textit{Haste}, \textit{country}, \textit{Germany}\}$. This reasoning path is not ***interpretable*** since we only know *Haste* is in *Germany*, but we have no idea whether *Haste* is a municipality or a district of *Germany*. However, from reasoning path $p_2 = \{\textit{Haste}, \textit{located in the administrative territorial entity}, \textit{Schaumburg}, \textit{country}, \textit{Germany}\}$, we can know *Haste* is a smaller administrative region than *Schaumburg*– possibly a municipality. Besides, compared to p_1, p_2 is more ***informative***. With p_1, to generate s, machines need to "guess" the district of *Haste*. However, with p_2, machines can predict the district of *Haste* is *Schaumburg* with a high confidence.

A more interpretable and informative reasoning path can guide and help machines to generate a more reasonable and precise relation description with the desired characteristics. This is because machines can more easily reason between entities with the path and incorporate more important information from the path. Therefore, instead of using the shortest paths, we design a mechanism to select the most interpretable and informative reasoning paths automatically. We achieve this by the confidence estimation of PLMs, which is motivated by related work on machine translation and speech recognition for accessing the quality of the prediction (Siu and Gish, 1999; Ueffing and Ney, 2007; Niehues and Pham, 2019). Given an entity pair (x, y), with a reasoning path $p(x, y)$, a trained model $\mathcal{M}$, and the corresponding prediction $\mathcal{M}(p(x, y))$, the confidence of the prediction can be evaluated by the posterior probability $P(\mathcal{M}(p(x, y))|p(x, y))$[3]. We select the reasoning path associated with the highest confidence score:

$$\hat{p}(x, y) = \underset{p(x,y)\in\mathcal{P}(x,y)}{\arg\max} P(\mathcal{M}(p(x, y))|p(x, y)).$$

Reasoning path selection by confidence estimation is intuitive since 1) if a reasoning path is more ***interpretable***, which means the path is easier to convert to a precise relation description, PLMs can "reason" between entities based on their knowledge with less effort; 2) if a reasoning path is more ***informative***, which means the reasoning path provides useful relational knowledge, PLMs can incorporate such information into the prediction without guessing the necessary information. In both cases, the confidence of the prediction will be higher.

With the reasoning path selection mechanism, given an entity pair (x, y), the generated relation description is $\mathcal{M}(\hat{p}(x, y))$, where $\hat{p}(x, y)$ is the reasoning path associated with the highest confidence score. The selected reasoning path can also serve as a support of the prediction and help users to understand the relation in a structured view. To get the trained model $\mathcal{M}$, we can directly apply *RelationBART-SP* introduced in Section 2.3. We name RelationBART-SP with reasoning path selection as **RelationBART-SP + PS**[4] To make the training more robust and let PLMs learn more features from valid reasoning paths, for each entity pair, we can sample more than one reasoning path, e.g., the shortest n reasoning paths with hops $\leq k$, to train the model. We refer to this model as **RelationBART-MP + PS**.

3 Experiments

3.1 Dataset Construction and Analysis

We use Wikipedia and Wikidata (Vrandečić and Krötzsch, 2014) to build a benchmark dataset for open relation modeling.

[2] If there exist multiple shortest paths for an entity pair, we randomly choose one.

[3] We use the posterior probability of BART implemented by `fairseq`. The estimation may be further improved through calibration (Jiang et al., 2021).

[4] We encourage the model to select relatively short paths since long paths are likely to introduce redundant information and the reasoning will not be intuitive, e.g., {*Haste*, *shares border with*, *Hohnhorst*, *shares border with*, *Bad Nenndorf*, *country*, *Germany*}.

	train	dev	test	test*
number	5,434,158	27,431	55,226	7,302

	1-hop	2-hop	3-hop	$>$ 3-hop
ratio (%)	35.14	17.80	7.33	39.73

Table 1: The statistics of the data.

The first sentences of Wikipedia are definition-like sentences connecting different entities. For instance, the first sentence of page *Deep Learning* is $s =$ "*[Deep learning] (also known as deep structured learning) is part of a broader family of [machine learning] methods based on [artificial neural networks] with [representation learning].*" The head entity of this sentence is *deep learning*, and there are three tail entities: *machine learning*, *artificial neural networks*, and *representation learning*, which are linked to other pages and can be easily extracted with simple text preprocessing. Combining the head entity and the three tail entities, we can construct three entity pairs, whose expected relation descriptions are all s. The version we used is 2021-03-20 dump[5] of English Wikipedia. For each page, we extract the plain text by WikiExtractor[6] and further extract the first sentence. We randomly split entity pairs to build train/dev/test sets, where the head entities *do not* overlap in each set.

To provide reasoning paths for open relation modeling, we sample part of Wikidata to build a knowledge graph. Specifically, we keep facts whose head and tail entities both appear in Wikipedia. The extracted KG contains 5,033,531 entities and 23,747,210 fact triples. The relation between two entities is considered as k-hop if the shortest reasoning path between them is k-hop.

Analysis and Filtering. To assess the quality of the dataset, we randomly sample 100 examples from the test set and ask human annotators to judge whether each sentence well represents entity relationships. As a result, 87% of the sentences are considered as good relation descriptions.

To improve the quality of evaluation, we design a rule-based method to construct a high-quality sub-test set. Specifically, we collect dependency graph for each relation description, and calculate the *dependency coverage*: the ratio of tokens covered by the shortest dependency path from the head to the tail compared to all the tokens in the sentence; and *surface coverage*: the ratio of tokens between the head and the tail (including head and tail) compared to all the tokens in the sentence. For instance, given entity pair (*Walton East*, *parish*) and relation description "*Walton East is a small rural village and parish established around a church at least as early as Norman times.*" The shortest dependency path from the head to the tail only contains tokens {Walton, East, is, parish}, so the *dependency coverage* is $4/20$. And there are 9 tokens between the head and tail, so the *surface coverage* is $9/20$.

A low *dependency coverage* and *surface coverage* indicate that many tokens in the sentence may not be important to characterize the relation between the head and the tail; therefore, the sentence may not be a good relation description. We keep examples whose $(\text{dependency coverage} + \text{surface coverage})/2 > 0.6$. After filtering, 96% of the sentences are judged as good relation descriptions by the human annotators. Here we note that while the above method filters out bad examples, it also filters out many good relationship descriptions. Table 1 summarizes the statistics of the data (test* denotes the filtered sub-test set).

3.2 Experimental Setup

Baselines. Because our task on open relation modeling is new, there is no existing baseline for model comparison. We design the following baselines/variants for evaluation:

- **DefBART**: Since the expected output is a definition-like sentence, the model proposed in (Bevilacqua et al., 2020) can be applied directly, i.e., generating the definition of the head entity with the head entity as input. We can observe the performance gain of relation modeling compared to definition modeling in terms of generating definitions and see the difference between them.
- **RelationBART-Vanilla**: The vanilla version of our model introduced in Section 2.2.
- **RelationBART-SP**: The shortest-path version of our model introduced in Section 2.3.
- **RelationBART-SP + PS**: The shortest-path version of our model, combining with the reasoning path selection mechanism (Section 2.4).
- **RelationBART-MP + PS**: The multiple-path version of our model, combining with the reasoning path selection mechanism (Section 2.4).

Without additional notation, we apply the BART-base model and denote "Large" when using the BART-large model. "w/o PT" means the BART-

[5] https://dumps.wikimedia.org/enwiki/20210320

[6] https://github.com/attardi/wikiextractor

	BL	R-L	MT	BS
DefBART	20.67	41.82	18.84	81.56
RelationBART-Vanilla (w/o PT)	26.01	50.84	23.65	85.37
RelationBART-SP (w/o PT)	26.60	51.86	24.15	85.79
RelationBART-SP (w/o PT) + PS	27.60	52.70	24.75	85.99
RelationBART-MP (w/o PT) + PS	**28.75**	**53.46**	**25.34**	**86.43**
RelationBART-Vanilla	26.81	51.48	24.14	85.73
RelationBART-SP	27.78	52.59	24.79	86.20
RelationBART-SP + PS	28.83	53.48	25.42	86.40
RelationBART-MP + PS	**29.51**	**53.74**	**25.64**	**86.51**
RelationBART-Vanilla (Large)	27.93	52.10	24.72	86.03
RelationBART-SP (Large)	29.21	53.01	25.37	86.43
RelationBART-SP (Large) + PS	**30.31**	53.85	**25.99**	86.61
RelationBART-MP (Large) + PS	29.72	**54.10**	25.89	**86.70**

Table 2: Results of open relation modeling on the full test set (test).

	BL	R-L	MT	BS
DefBART	25.98	47.38	22.39	83.41
RelationBART-Vanilla (w/o PT)	34.70	59.57	28.85	88.01
RelationBART-SP (w/o PT)	35.48	60.55	29.40	88.43
RelationBART-SP (w/o PT) + PS	38.62	62.60	31.07	89.05
RelationBART-MP (w/o PT) + PS	**40.52**	**63.73**	**32.06**	**89.53**
RelationBART-Vanilla	35.45	59.92	29.33	88.25
RelationBART-SP	36.58	61.15	30.04	88.75
RelationBART-SP + PS	39.93	63.32	31.80	89.39
RelationBART-MP + PS	**41.43**	**64.15**	**32.45**	**89.64**
RelationBART-Vanilla (Large)	36.53	60.54	29.90	88.50
RelationBART-SP (Large)	37.65	61.34	30.57	88.89
RelationBART-SP (Large) + PS	41.21	63.56	32.41	89.53
RelationBART-MP (Large) + PS	**41.46**	**64.36**	**32.62**	**89.79**

Table 3: Results of open relation modeling on the filtered test set (test*).

base model is not pre-trained.

Metrics. Following existing works on text generation, we apply several widely-used metrics to automatically evaluate the performance of open relation modeling, including BLEU (Papineni et al., 2002), ROUGE-L (Lin, 2004), METEOR (Banerjee and Lavie, 2005), and BERTScore (Zhang et al., 2019). Among them, BLEU (BL) and ROUGE-L (R-L) are based on simple string matches, and METEOR (MT) also incorporates word stems, synonyms, and paraphrases for matching. These three metrics mainly focus on measuring surface similarities. BERTScore (BS) is based on the similarities of contextual token embeddings. We also conduct human evaluation by asking three human annotators to assign graded values (1-4) to the sampled predictions according to Table 8.[7]

3.3 Open Relation Modeling

Tables 2 and 3 summarize the experimental results of open relation modeling with the automatic metrics. We observe that RelationBART-Vanilla achieves much better performance than DefBART, which demonstrates the necessity of the tail entity in terms of generating definition-like sentences that imply relations between entities. Besides, RelationBART variants outperform the versions without pre-training, which indicates that knowledge stored in PLMs after pre-training is helpful for open relation modeling. However, the improvement is not significant, which may be because the size of our training data is large; thus the model can learn rich knowledge about entities from definitions without pre-training. To verify this, we also train the model with smaller sizes of data in Appendix B.

hard-to-reason ($>$ 3-*hop*)	BL	R-L	MT	BS
RelationBART-Vanilla	22.99	47.25	22.21	84.39
RelationBART-SP	**23.07**	**47.36**	**22.32**	**84.42**
RelationBART-SP + PS	**23.07**	**47.36**	**22.32**	**84.42**
RelationBART-MP + PS	22.63	46.91	21.99	84.24
RelationBART-Vanilla (Large)	24.24	**47.97**	22.88	**84.76**
RelationBART-SP (Large)	**24.50**	47.81	**22.90**	84.70
RelationBART-SP (Large) + PS	**24.50**	47.81	**22.90**	84.70
RelationBART-MP (Large) + PS	22.92	47.45	22.34	84.55
reasonable ($\leq$ 3-*hop*)	**BL**	**R-L**	**MT**	**BS**
RelationBART-Vanilla	29.61	54.25	25.56	86.61
RelationBART-SP	31.24	56.00	26.62	87.35
RelationBART-SP + PS	33.04	57.48	27.73	87.70
RelationBART-MP + PS	**34.52**	**58.21**	**28.36**	**87.99**
RelationBART-Vanilla (Large)	30.64	54.81	26.08	86.86
RelationBART-SP (Large)	32.66	56.42	27.20	87.56
RelationBART-SP (Large) + PS	34.55	57.81	28.29	87.85
RelationBART-MP (Large) + PS	**34.69**	**58.45**	**28.54**	**88.11**

Table 4: Results of open relation modeling for *reasonable* and *hard-to-reason* pairs.

Compared to RelationBART-Vanilla, the models with reasoning paths all achieve better performance, which demonstrates that reasoning paths can help PLMs reason between entities and provide additional relational knowledge to PLMs for open relation modeling. Besides, the models with reasoning path selection mechanism outperform the ones without it, which indicates PLMs can select more interpretable and informative reasoning paths based on confidence estimation, and the selected reasoning paths can guide PLMs to generate more reasonable and precise relation descriptions.

We also divide the testing examples into two groups: *reasonable*, where the entities can be reasoned within 3 hops in the Wikidata knowledge graph, and *hard-to-reason*, where the entities cannot be reasoned within 3 hops. From the results shown in Table 4, we observe that, for the *reasonable* pairs, the performance improvement is signif-

[7]Details about implementation are in Appendix A.

	Rating (1-4)
RelationBART-Vanilla (Large)	2.67
RelationBART-SP (Large)	2.82
RelationBART-MP (Large) + PS	**3.01**

Table 5: Qualitative results of open relation modeling.

	Accuracy (%)
Random Walk	64.43
Shortest Path	61.34
RelationBART-SP (Large)	72.68
RelationBART-MP (Large)	**80.93**

Table 6: Results of reasoning path selection.

icant, while for the *hard-to-reason* pairs, there is not much difference in model performance. This is because, for *hard-to-reason* pairs, PLMs cannot incorporate additional relational knowledge from KGs only with encoding "x; *unknown:* y"– which shows the training of the model is stable and the variance of the results is low. Besides, all the models perform much better on *reasonable* pairs, which indicates if two entities can be reasoned in existing KGs with fewer hops, it is easier to generate their relation descriptions with PLMs, no matter whether a reasoning path is incorporated or not.

Qualitative Evaluation. We also perform a qualitative evaluation by asking three annotators to assign graded values to relation descriptions generated by our models according to Table 8. We randomly sample 100 *reasonable* entity pairs from the test set for evaluation. The average pairwise Cohen's kappa is 0.67, which indicates a substantial agreement (0.61-0.8) (Landis and Koch, 1977).

From Table 5, we observe the performance is satisfactory. Our best model *RelationBART-MP (Large) + PS* achieves a rating of about 3, which means the model can often generate a relation description that well captures the relation, where only minor errors that do not affect the understanding of the relation are included. In addition, the qualitative evaluation results are consistent with the quantitative evaluation results in Table 2 and Table 4, which validates the function of reasoning paths and reasoning path selection mechanism.

3.4 Reasoning Path Selection

Results in Tables 2, 3, 4, and 5 indicate machines can select better reasoning paths for open relation modeling by confidence estimation. We also test the quality of the selected reasoning paths from a human understanding perspective.

We randomly select 300 entity pairs from the test set and ensure all the pairs are associated with at least two reasoning paths with hops ≤ 3. For each entity pair, we randomly select 2 reasoning paths and manually label which one is more interpretable and informative, i.e., humans can understand the relation between two entities more easily by reading the reasoning path. We skip pairs that are difficult to judge which path is better. Among the 300 pairs, 106 pairs were skipped.

Table 6 reports the results of reasoning path selection with different methods. The *Random Walk* baseline selects the reasoning path by the probability of generating the path starting from the head entity, which is suggested by (Lao et al., 2011). The *Shortest Path* baseline selects the path with a shorter length (for 52 cases where the length of two paths is the same, we randomly choose one).

We can see the performance of RelationBART-MP (Large) is quite impressive, where machines make the same choices as humans in more than 80% of the cases. In addition, results in Table 6 are consistent with results in Table 2, which indicates a better reasoning path selection mechanism can promote machines to generate better relation descriptions.

3.5 Generation Examples and Error Analysis

Table 7 shows some generation examples via the RelationBART-MP (Large) model. The first row of each group is the reference definition in Wikipedia. 1) The first group contains reasoning paths connecting *Romeries* and *France*, we can see the second reasoning path is the most interpretable and informative, where the confidence score is the highest, and the generated relation description is the same as the ground truth. 2) The second group contains four entity pairs whose head entity is *data mining*, and the reasoning path in each row is the one associated with the highest confidence score. By reading these generated relation descriptions, humans can better understand *data mining* and its relationship with other terms. 3) The third group contains five *hard-to-reason* entity pairs whose head entity is *evaluation*. We can see the reference definition of *evaluation* is quite abstract that cannot capture the relation between *evaluation* and a specific field, while by reading the generated ones, humans can understand what *evaluation* means in different fields and how it relates to them.

Input	Reasoning Path (Encoding)	Output	Score
Romeries **[reference]**	-	Romeries is a commune in the Nord department in northern France.	-
(Romeries, France)	Romeries; country: France **[NOT selected]**	Romeries is a commune in the Somme department in Hauts-de-France in northern France.	-0.29
	Romeries; located in the administrative territorial entity: Nord; country: France **[selected]**	Romeries is a commune in the Nord department in northern France.	-0.13
	Romeries; shares border with: Solesmes; country: France **[NOT selected]**	Romeries is a commune in the Marne department in north-eastern France.	-0.31
data mining **[reference]**	-	Data mining is a process of extracting and discovering patterns in large data sets involving methods at the intersection of machine learning, statistics, and database systems.	-
(data mining, artificial intelligence)	data mining; facet of: machine learning; subclass of: artificial intelligence	Data mining is a subfield of machine learning and artificial intelligence concerned with the collection, processing, and analysis of large amounts of data.	-0.95
(data mining, database)	data mining; facet of: database	Data mining is the process of extracting information from a data set, such as a database.	-0.94
(data mining, statistics)	data mining; facet of: machine learning; partially coincident with: statistics	Data mining is the process of collecting, processing, and analyzing large amounts of data in the form of statistics and machine learning.	-1.04
(data mining, data)	data mining; facet of: machine learning; partially coincident with: statistics; studies: data	Data mining is the process of extracting useful information from data.	-0.81
evaluation **[reference]**	-	Evaluation is a systematic determination of a subject's merit, worth and significance, using criteria governed by a set of standards.	-
(evaluation, machine learning)	evaluation; unknown: machine learning	In computer science, evaluation is the process of evaluating a machine learning algorithm to determine whether the algorithm is performing well.	-1.30
(evaluation, algorithm)	evaluation; unknown: algorithm	In computer science, evaluation is the process of evaluating an algorithm to determine whether it is correct.	-1.13
(evaluation, robotics)	evaluation; unknown: robotics	In robotics, evaluation is the process of determining whether or not a particular component of a system is working properly.	-1.54
(evaluation, software engineering)	evaluation; unknown: software engineering	In computer science and software engineering, evaluation is the process of determining whether a particular feature or feature should be added to a product or service.	-1.26
(evaluation, computer security)	evaluation; unknown: computer security	In computer security, evaluation is the process of determining the security of a computer system.	-1.09
The Association for Computational Linguistics **[reference]**	-	The Association for Computational Linguistics (ACL) is the international scientific and professional society for people working on problems involving natural language and computation.	-
(The Association for Computational Linguistics, natural language processing)	The Association for Computational Linguistics; unknown: natural language processing	The Association for Computational Linguistics (ACL) is a professional association in the field of natural language processing (NLP).	-0.60
(The Association for Computational Linguistics, artificial intelligence)	The Association for Computational Linguistics; unknown: artificial intelligence	The Association for Computational Linguistics (ACL) is a professional association for linguists working in the field of computational linguistics, including artificial intelligence, machine learning, natural language processing, and computational linguistics.	-0.67

Table 7: Sample of relation descriptions generated by RelationBART-MP (Large).

Error Analysis. To further understand the quality of the outputs produced by our model and identify the remaining challenges, we investigate the error cases found by examining the generated relation descriptions. As a result, we found most errors can refer to as *hallucinations*, i.e., producing irrelevant or contradicted facts. This type of error is mainly due to knowledge coming from pre-training, fine-tuning, and reasoning paths is not sufficient.

Taking entity pair $(\textit{Romeries}, \textit{France})$ in Table 7 as an example, if the model takes the shortest reasoning path, i.e., *Romeries; country: France*, as input, a relation description that wrongly predicts the department of *Romeries* will be generated. This is because knowledge about the department is missing from the reasoning path, and such detailed knowledge is also difficult to obtain from the parameters of the trained model.

Another example is $(\textit{Play It Loud}, \textit{rock music})$, where the reference relation description is "*Play It Loud is the second studio album by the British rock group Slade.*" The reasoning path selected by RelationBART-MP (Large) is $\{\textit{Play It Loud}, \textit{performer}, \textit{Slade}, \textit{genre}, \textit{hard rock}, \textit{subclass of}, \textit{rock music}\}$. This reasoning path contains detailed knowledge about the performer; however, it is still difficult to judge whether *Play It Loud* is a song or an album. As a result, the model generates "*Play It Loud is a song by the British rock band Slade.*"

Hallucination is a common issue and challenging problem in text generation. From the results in Table 5 and the generation examples, we can observe hallucination is reduced by incorporating reasoning paths and the reasoning path selection mechanism. How to further alleviate it for open relation modeling will be our further work direction. We discuss some possible solutions in Section 4.

4 Discussion

Limitation of Definitional Sentences. Although a considerable number of relations can be well captured by definitional sentences, there are types of relations that are not natural to be represented by definitional sentences. For instance, for *Kobe Bryant* and *Shaq O'neal* (both are NBA players in Los Angeles Lakers), it is not natural to assume one would appear in the other's definition. In this case, we can include a third related entity to help users to understand their relation. For example, we can include *Los Angeles Lakers* (which can be found from a knowledge graph or a corpus); and then, we can generate two sentences: 1) "Kobe Bryant was an NBA player in Los Angeles Lakers"; 2) "Shaq O'neal was an NBA player in Los Angeles Lakers". With these two sentences, users can easily understand their relation. It is also possible to design

a model to synthesize these two sentences to one (Becker et al., 2021), e.g., "Kobe Bryant and Shaq O'neal were both NBA players played in Los Angeles Lakers". We leave a comprehensive solution to solve this limitation as future work.

Open Relation Modeling with Diversity. In the real world, multiple important relations can be associated with one entity pair. Considering this, as future work, we may generate diverse relation descriptions for one entity pair with different reasoning paths selected.

Open Relation Modeling with More Knowledge. Open relation modeling is a knowledge-intensive task (Lewis et al., 2020b), where knowledge about entities and relations is essential to solving this task. In this work, we incorporate knowledge from model pre-training, definitions of entities, and reasoning paths. The proposed model can achieve impressive performance, especially for reasonable entity pairs. As future work, we can leverage more external information of entities, e.g., sentences/paragraphs containing the target entities from corpora, to provide more knowledge for open relation modeling.

5 Related Work

Previously, Voskarides et al. (2015) study the problem of extracting sentences that describe relations between entities with direct relations in a knowledge graph. They model this task as a learning to rank problem and design a supervised learning model with manually annotated sentences. As follow-up work, Huang et al. (2017) solve this task with training data built by leveraging clickthrough data from Web search, and Voskarides et al. (2017) generate the description of a relationship instance in a knowledge graph by filling created sentence templates with appropriate entities. The ability of these models is limited since they heavily rely on features of entities and relations; thus these models can only handle entities with several pre-specified types (only 10 in (Voskarides et al., 2017)) of explicit relations in KGs (e.g., *isMemberOfMusicGroup*), while our methods can deal with a large number of types of relations, including implicit ones (e.g., *evaluation* and *algorithm*), i.e., in an "open" setting.

Recently, Lin et al. (2020); Liu et al. (2021b) study a constrained text generation problem that aims to generate coherent sentences describing everyday scenarios containing the given common concepts. Different from them, we aim to generate sentences that can explain the relation between entities intuitively and explicitly. Dognin et al. (2020); Agarwal et al. (2021) study the data-to-text generation problem (Kukich, 1983) that converts the KG into natural text with language models. The focus of these works is to convert knowledge graphs into natural language, while we propose to discover relation descriptions between entities with pre-trained language models. Besides, only common concepts or entities with direct relations are studied in these works, while our methods deal with entities with multi-hop relations, even including entities that cannot be reasoned in existing KGs.

6 Conclusion

In this paper, we introduce and study the novel *open relation modeling* problem– generating coherent sentences describing general relations between entities, where the relations can be multi-hop, even cannot be reasoned in an existing KG. We achieve this by teaching PLMs to learn from defining entities and select/utilize reasoning paths. We believe this work will open a door for modeling relations between entities. As for future work, we plan to improve our model as discussed in Section 4 and apply our methods to downstream applications, e.g., a system for users to explore relations between entities, which can be further applied to explore a taxonomy or ontology. We can also use the generated relation descriptions to help some related tasks, such as relation extraction (Bach and Badaskar, 2007), knowledge graph construction and completion (Ji et al., 2021). The trained models can be further fine-tuned for open relation modeling on specific domains.

Acknowledgements

We thank the reviewers for their constructive feedback. This material is based upon work supported by the National Science Foundation IIS 16-19302 and IIS 16-33755, Zhejiang University ZJU Research 083650, IBM-Illinois Center for Cognitive Computing Systems Research (C3SR)– a research collaboration as part of the IBM Cognitive Horizon Network, grants from eBay and Microsoft Azure, UIUC OVCR CCIL Planning Grant 434S34, UIUC CSBS Small Grant 434C8U, and UIUC New Frontiers Initiative. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the funding agencies.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565.

Nguyen Bach and Sameer Badaskar. 2007. A review of relation extraction. *Literature review for Language and Statistics II*, 2:1–15.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJCAI*.

Maria Becker, Siting Liang, and Anette Frank. 2021. Reconstructing implicit knowledge with language models. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 11–24.

Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. Generationary or:"how we went beyond word sense inventories and learned to gloss". In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7207–7221.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris 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*, volume 33, pages 1877–1901. Curran Associates, Inc.

Wenhu Chen, Wenhan Xiong, Xifeng Yan, and William Wang. 2018. Variational knowledge graph reasoning. *arXiv preprint arXiv:1803.06581*.

Pierre Dognin, Igor Melnyk, Inkit Padhi, Cicero dos Santos, and Payel Das. 2020. Dualtkb: A dual learning bridge between text and knowledge base. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8605–8616.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *CogSci*.

Jizhou Huang, Wei Zhang, Shiqi Zhao, Shiqiang Ding, and Haifeng Wang. 2017. Learning to explain entity relationships by pairwise ranking with convolutional neural networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 4018–4025.

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*.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Karen Kukich. 1983. Design of a knowledge-based report generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

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*, pages 529–539.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. Commongen: A constrained text generation challenge for generative commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1823–1840.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021a. Gpt understands, too. *arXiv preprint arXiv:2103.10385*.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and S Yu Philip. 2021b. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6418–6425.

Jan Niehues and Ngoc-Quan Pham. 2019. Modeling confidence in sequence-to-sequence models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 575–583.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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)*, pages 2463–2473.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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. *Journal of Machine Learning Research*, 21:1–67.

Kyle Richardson and Ashish Sabharwal. 2020. What does my qa model know? devising controlled probes using expert knowledge. *Transactions of the Association for Computational Linguistics*, 8:572–588.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426.

Michael Schmitz, Stephen Soderland, Robert Bart, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 523–534.

Manhung Siu and Herbert Gish. 1999. Evaluation of word confidence for speech recognition systems. *Computer Speech & Language*, 13(4):299–319.

Nicola Ueffing and Hermann Ney. 2007. Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40.

Nikos Voskarides, Edgar Meij, and Maarten de Rijke. 2017. Generating descriptions of entity relationships. In *European Conference on Information Retrieval*, pages 317–330. Springer.

Nikos Voskarides, Edgar Meij, Manos Tsagkias, Maarten De Rijke, and Wouter Weerkamp. 2015. Learning to explain entity relationships in knowledge graphs. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 564–574.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Chenguang Wang, Xiao Liu, and Dawn Song. 2020. Language models are open knowledge graphs. *arXiv preprint arXiv:2010.11967*.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. *arXiv preprint arXiv:1707.06690*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pre-trained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9733–9740.

Rating	Criterion
4	The relation is well captured, and important information about entities is included and correctly predicted.
3	The prediction contains minor error(s) that do not affect the understanding of the relation.
2	The prediction contains major error(s) that affect the understanding of the relation, while the relation can still be inferred to some extent.
1	The prediction contains major error(s) that will mislead the understanding of the relation.

Table 8: Annotation guidelines excerpt.

A Implementation Details

We employ the `fairseq` library[8] to build the RelationBART model and adopt the key hyperparameters as suggested in (Lewis et al., 2020a). We manually set the learning rate as 5×10^{-5} and batch-size of 1,024 tokens based on some preliminary experiments and the memory size of GPUs. We set the maximum reasoning length as 3 since the number of reasoning paths with hops > 3 is very large and the quality of these paths is generally low. For RelationBART-MP and reasoning path selection, we sample at most 5 reasoning paths with hops ≤ 3. All the models were trained on NVIDIA Quadro RTX 5000 GPUs, and the training converged in 50 epochs. The training time of RelationBART-Vanilla, RelationBART-MP, and RelationBART-MP (Large) for one epoch with 3 GPUs are 80 minutes, 4 hours, and 7 hours respectively.

B Open Relation Modeling with Different Sizes of Training Data

100%	**BL**	**R-L**	**MT**	**BS**
RelationBART-Vanilla (w/o PT)	26.01	50.84	23.65	85.37
RelationBART-Vanilla	26.81	51.48	24.14	85.73
10%	**BL**	**R-L**	**MT**	**BS**
RelationBART-Vanilla (w/o PT)	22.88	48.50	22.07	84.31
RelationBART-Vanilla	24.31	49.89	22.99	85.16
1%	**BL**	**R-L**	**MT**	**BS**
RelationBART-Vanilla (w/o PT)	17.30	44.12	19.02	81.56
RelationBART-Vanilla	20.99	47.11	21.23	84.04

Table 9: Results of open relation modeling with *100%*, *10%*, and *1%* training data.

From Table 9, we observe that when the training data become smaller, the performance of the version without pre-training decreases much faster than the one with pre-training.

[8] https://github.com/pytorch/fairseq/tree/master/examples/bart