Learning Reasoning Patterns for Relational Triple Extraction with Mutual Generation of Text and Graph

Yubo Chen[*] **Yunqi Zhang**[*] **Yongfeng Huang**
Department of Electronic Engineering & BNRist, Tsinghua University, Beijing, China
ybch14@gmail.com yq-zhang19@mails.tsinghua.edu.cn
yfhuang@tsinghua.edu.cn

Abstract

Relational triple extraction is a critical task for constructing knowledge graphs. Existing methods focused on learning text patterns from explicit relational mentions. However, they usually suffered from ignoring relational reasoning patterns, thus failed to extract the implicitly implied triples. Fortunately, the graph structure of a sentence's relational triples can help find multi-hop reasoning paths. Moreover, the type inference logic through the paths can be captured with the sentence's supplementary relational expressions that represent the real-world conceptual meanings of the paths' composite relations. In this paper, we propose a unified framework to learn the relational reasoning patterns for this task. To identify multi-hop reasoning paths, we construct a relational graph from the sentence (text-to-graph generation) and apply multi-layer graph convolutions to it. To capture the relation type inference logic of the paths, we propose to understand the unlabeled conceptual expressions by reconstructing the sentence from the relational graph (graph-to-text generation) in a self-supervised manner. Experimental results on several benchmark datasets demonstrate the effectiveness of our method.

1 Introduction

Relational triple extraction is defined as automatically recognizing semantic relations with triple structures (*subject*, *relation*, *object*) among multiple entities in a sentence. It is a critical task for natural language processing, especially for Knowledge Graph (KG) construction from unlabeled corpus (Dong et al., 2014).

Recent work proposed several neural network methods to extract relational triples. For example, Zheng et al. (2017) proposed a sequence tagging scheme for this task but failed to extract overlapping triples. Wei et al. (2020) proposed to solve the overlapping triple problem with a binary tagging framework. Zeng et al. (2018) proposed to address this issue by generating triple element sequences with copy mechanism.

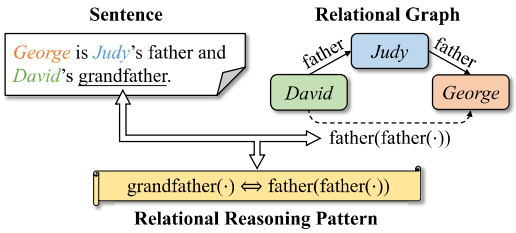

Figure 1: An example of the relational graph and the relational reasoning pattern. Solid arrows of the relational graph are golden relational triples. The dashed arrow is a two-hop reasoning path.

Existing methods achieved considerable success in learning text patterns of relational triples from explicit mentions. However, they usually suffered from the failure of extracting the relational triples which are implicitly implied in the text (Zhu et al., 2019). This is because they ignored relational reasoning patterns in natural language, which usually consist of *finding multi-hop paths* and *inferring relation types along these paths*. For example, in Figure 1, the triple ("*David*", "*father*", "*Judy*") is not explicitly expressed in the sentence and requires relational reasoning to be extracted. Unfortunately, the ignorance of relational reasoning patterns in existing methods will cause serious incompleteness of the constructed KGs and performance degradation of downstream tasks (Angeli and Manning, 2013; Jia et al., 2020).

Our work is motivated by several observations. First, the relational triples of a sentence usually have a graph structure, which is useful for finding multi-hop reasoning paths. For example, in Figure 1, the relational graph provides a two-hop reasoning path between "*David*" and "*George*". Second, the sentence usually contains supplementary relational expressions that represent the real-world

[*]Equal contribution.

Findings of the Association for Computational Linguistics: ACL 2022, pages 1638 - 1647

conceptual meanings of the paths' composite relations, which can help capture the relation type inference logic through the paths. For example, in Figure 1, the phrase "*'s grandfather*" helps capture the equivalence between the composite relation "*father(father(·))*" and the real-world relational concept "*grandfather*", which reflects the relation type inference logic of the two-hop path.

In this paper, we propose a unified framework to learn reasoning patterns for the relational triple extraction task. First, we construct a relational graph from the sentence, i.e. **text-to-graph** generation, to identify potential multi-hop reasoning paths. Then we utilize a multi-layer Relational Graph Convolution Network (R-GCN) (Schlichtkrull et al., 2018) to propagate node information along these paths. Next, to capture the relation type inference logic of the reasoning paths, we aim to exploit and understand the conceptual expressions in the sentence, but the absence of human annotations for these expressions poses a huge challenge. To tackle this challenge, we propose a self-supervised reconstruction of the sentence from the relational graph, i.e. **graph-to-text** generation. Our model captures the relation type inference logic by learning to recover the conceptual expressions from the symbolic relation composition, such as the recovery of "*'s grandfather*" from "*father(father(·))*" in Figure 1. Finally, we use the reasoning pattern enhanced model to extract relational triples from the sentence.

The main contributions of this paper are:

- We propose a mutual generation framework of text and graph to learn relational reasoning patterns for relational triple extraction.
- To identify multi-hop reasoning paths, we construct a relational graph from the sentence and apply a multi-layer R-GCN to the graph.
- To capture the relation type inference logic of the paths, we propose to exploit the unlabeled conceptual expressions with a self-supervised sentence reconstruction task from the graph.
- Experimental results on several datasets indicate the effectiveness of our method.

2 Related Work

Early work extracted relational triples with pipeline systems (Zelenko et al., 2003; Zhou et al., 2005; Chan and Roth, 2011; Gormley et al., 2015), but they usually suffered from error propagation problems. Also, they failed to capture the interactions between entities and relations. To address these issues, jointly extracting entities and relations with an end-to-end model has become the main paradigm of this task. Previous work proposed several feature-based models (Yu and Lam, 2010; Li and Ji, 2014; Ren et al., 2017). For example, Ren et al. (2017) proposed a joint embedding framework to map entities, relations, text features and type labels into unified low-dimensional spaces. Afterward, several neural network-based methods were proposed to eliminate hand-crafted features (Gupta et al., 2016; Miwa and Bansal, 2016; Zheng et al., 2017). For example, Zheng et al. (2017) proposed to extract relational triples directly with a sequence tagging model, whose tags contain the information of entities and the relations they hold. However, they assigned only one label to each word and failed to extract multiple triples whose entities overlap with each other.

Recent work proposed several mechanisms to address the overlapping triple problem, such as sequence tagging variations (Wei et al., 2020; Wang et al., 2020; Zheng et al., 2021) and triple element generation (Zeng et al., 2018, 2019, 2020; Sui et al., 2020; Huguet Cabot and Navigli, 2021). For example, Wei et al. (2020) proposed a cascade binary tagging framework and modeled relations as functions that map subjects to objects. Zheng et al. (2021) proposed to decompose the task into three subtasks: relation judgment, entity extraction and subject-object alignment. Zeng et al. (2018) proposed to generate the element sequence of triples with a copy-based seq2seq model, while Sui et al. (2020) proposed to generate the set of triples with a set prediction network. However, these methods mainly focused on learning text patterns of the explicitly mentioned triples. They usually ignored the relational reasoning patterns thus failed to extract the implicitly implied triples (Zhu et al., 2019). Although Chen et al. (2021) proposed a reasoning pattern enhanced model, they utilized entity type information, which requires extra supervision.

Different from previous work, we propose a mutual generation framework of text and graph to capture relational reasoning patterns. We identify multi-hop reasoning paths by generating a relational graph from the sentence. We propose to capture the relation type inference logic by incorporating supplementary conceptual expressions with self-supervised sentence generation from the graph. Experimental results on several datasets demonstrate the effectiveness of our method.

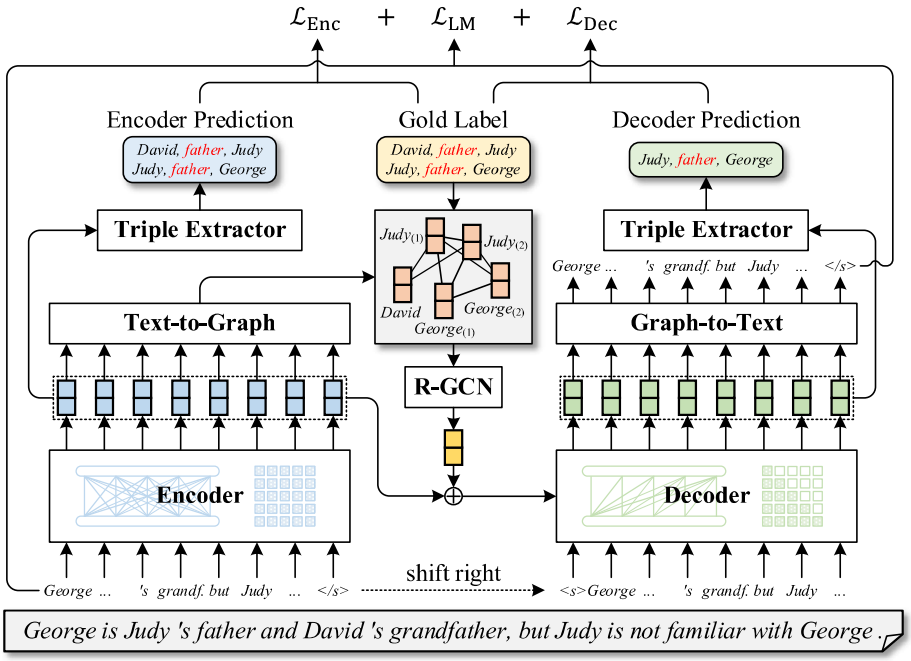

Figure 2: The overall framework of our approach. When recovering the sentence, we use the left-to-right Language Model (LM) objective, which is controlled by the lower triangular attention mask of the Transformer decoder.

3 Our Approach

The overall framework of our approach is illustrated in Figure 2. We introduce the text-to-graph and the graph-to-text generation methods in Section 3.1 and 3.2, respectively. Then we introduce the triple extractor in Section 3.3 and the details of training and inference in Section 3.4.

3.1 Text-to-Graph Generation

Relational reasoning in natural language is challenging because it usually requires reasoning for multiple hops. We observe that the graph structure of a sentence's relational triples can help identify multi-hop reasoning paths. Therefore, we construct a relational graph from the sentence to find multi-hop paths and apply multi-layer graph convolutions to propagate information along the paths.

First, we encode the words in the sentence into dense vector representations. Given the sentence $[x_1, \ldots, x_n]$, we employ a bi-directional Pre-trained Language Model (PLM) based on Transformers (Vaswani et al., 2017) as the encoder to capture the context of the sentence. We use the last hidden states $[\mathbf{h}_1^E, \ldots, \mathbf{h}_n^E]$ of the PLM as the contextual representations of the words.

Next, we use the word representations and the ground truth of relational triples to obtain the relational graph. We denote the graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_{|\mathcal{V}|}\}$ are the nodes with feature vectors, $\mathcal{R} = \{r_1, \ldots, r_{|\mathcal{R}|}\}$ are the relation types and $\mathcal{E} = \{(\mathbf{v}_i, r_k, \mathbf{v}_j), \ldots\}$ are the edges of the graph. We first utilize the text spans of the golden triples' entities to find the positions of all entity mentions in the sentence by perfect matching. We consider each entity mention $m = [x_{s_m}, \ldots, x_{e_m}]$ as a graph node, where s_m and e_m are the mention's start and end positions, respectively. We average the contextual word representations of the corresponding positions to obtain the feature vector $\mathbf{v} = \text{Average}([\mathbf{h}_{s_m}^E, \ldots, \mathbf{h}_{e_m}^E]) \in \mathcal{V}$. Then we add three kinds of edges to $\mathcal{E}$, as shown in Figure 3: (1) *Golden edges*, which connect all nodes (mentions) of the subject s and the object o with relation r for each golden triple (s, r, o). These edges provide the basic relation information of the golden triples. (2) *Reversed golden edges*, which are the reverse of the golden edges with new reverse relation types. These edges are added to allow sufficient bidirectional flow of node information to prevent some special graph structures from cutting off the information flow paths between nodes, such as siblings[1]. (3) *Co-reference edges*,

[1]For example, consider the graph B ← A → C. If reversed edges are not added, B and C will only be updated by A (and of course themselves) but A will never be updated by B or C. This will cut off the information flow path between B and C. If reversed edges are added, then each node can be updated by the other two, making the information flow more sufficient.

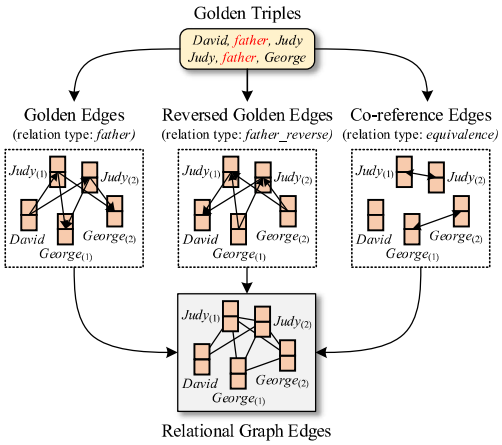

Figure 3: An example of the relational graph edges.

which connect all mentions pairs of the same entity with an equivalence relation. These edges are added to enhance entity representations (Wadden et al., 2019) because they propagate the rich information included in multiple mentions and their surrounding contexts. Therefore, the relation type set $\mathcal{R}$ contains the equivalence relation, the original relations of the dataset, and their reverse relations.

Finally, we employ an R-GCN (Schlichtkrull et al., 2018) with multiple layers to incorporate relation type information and propagate information along multi-hop paths. Following Guo et al. (2019), we add dense connections to the R-GCN. Formally, the convolution of the l-th layer is formulated as:

$$\mathbf{g}_i^{l+1} = \rho\Big(\mathbf{W}_s^l \mathbf{k}_i^l + \sum_{r\in\mathcal{R}} \sum_{j\in\mathcal{N}_i^r} \frac{1}{|\mathcal{N}_i^r|} \mathbf{W}_r^l \mathbf{k}_j^l\Big) \quad (1)$$

where ρ is an activation function (e.g. ReLU) and $\mathbf{W}_r^l$ and $\mathbf{W}_s^l$ are the transformation matrices of relation r and self-loops. $\mathcal{N}_i^r$ denotes neighbors of the i-th node under the relation r, and $\mathbf{k}_i^l = [\mathbf{g}_i^1, \ldots, \mathbf{g}_i^l]$ where $\mathbf{g}_i^1 = \mathbf{v}_i$. Then we feed the nodes' initial features and the R-GCN's output into a Multi-Layer Perceptron (MLP) and average the output to obtain the final graph representation: $\mathbf{g} = \text{Average}\big(\text{MLP}([\mathbf{v}; \mathbf{g}^L])\big)$.

3.2 Graph-to-Text Generation

Given a multi-hop reasoning path, inferring the relation type along the path is difficult because the inference logic usually reflects complicated commonsense facts. Fortunately, we observe that the sentence usually contains supplementary expressions that represent the real-world concepts of the paths' composite relations. These relational expressions can help capture the relation type inference logic. For example, in Figure 1, the symbolic composition of the two-hop relational path is "*father(father(·))*". The phrase "*'s grandfather*" in the sentence helps connect the composite relation and the real-world relational concept "*grandfather*", which reflects the fact that "*father*'s *father* is *grandfather*".

Based on this observation, we propose to exploit and understand the conceptual expressions in the sentence. However, the absence of human annotations for these concepts poses a great challenge. Inspired by self-supervised pre-training techniques of various PLMs (Devlin et al., 2019; Raffel et al., 2020; Lewis et al., 2020), we propose to reconstruct the sentence from the relational graph in a self-supervised manner to tackle this challenge. Our model learns the type inference logic by recovering the conceptual expressions from the symbolic relation compositions. For example, generating "*grandfather*" from "*father(father(·))*" represents the ability of understanding the logical equivalence between "*father*'s *father*" and "*grandfather*" (Radford et al., 2018; Tseng et al., 2020).

To reconstruct the sentence, we utilize an auto-regressive PLM as the decoder with the left-to-right LM objective. Given the sentence's encoder hidden states $[\mathbf{h}_{1:n}^E]$ and the graph representation $\mathbf{g}$, the standard graph-to-text decoder takes $\mathbf{g}$ and the right-shifted sentence $[\texttt{<s>}, x_1, \ldots, x_{n-1}]$ as input. However, we discover that the sentence may have relational irrelevant contents (e.g. "*is not familiar with*" in Figure 2), which may bring corruption to the reconstruction. To address this issue, we borrow part of the contextual information by feeding the average of $[\mathbf{h}_{1:n}^E]$ and $\mathbf{g}$ instead of $\mathbf{g}$ into the decoder. We denote the decoder's last hidden states as $[\mathbf{h}_{1:n}^D]$. Finally, we use a softmax classifier to predict the reconstructed tokens: $\mathbf{p}_i^{\text{LM}} = \text{softmax}\big(\mathbf{W}_d \mathbf{h}_i^D + \mathbf{b}_d\big)$.We choose the state-of-the-art T5 (Raffel et al., 2020) model as our backbone PLM because it has the same encoder-decoder structure as ours.

3.3 Triple Extractor

We employ CASREL (Wei et al., 2020) to extract relational triples. It consists of a subject tagger and relation-specific object taggers. The subject tagger first recognizes all possible subjects with two identical binary classifiers. It assigns each token a binary tag that indicates whether the current token corresponds to a subject's start or end position:

$$\mathbf{p}^{ss/se} = \sigma\big(\mathbf{W}_{ss/se}\mathbf{h} + \mathbf{b}_{ss/se}\big), \quad (2)$$

where σ is the sigmoid function, $\mathbf{h}$ are the input representations, $\mathbf{p}^{ss/se}$ are the probabilities of identifying all the tokens as the subject start/end positions, and $(\mathbf{W}_{ss}, \mathbf{b}_{ss}), (\mathbf{W}_{se}, \mathbf{b}_{se})$ are parameters of the two classifiers, respectively.

Then the relation-specific object taggers identify the objects and the involved relations w.r.t. the recognized subjects. Each object tagger corresponds to a relation type and has the same structure with the subject tagger. To incorporate the subject information, the object taggers take the averaged representation of the k-th subject's start and end tokens as $\mathbf{s}_k$ and predict the objects' start and end tags:

$$\mathbf{p}_{rk}^{os/oe} = \sigma\big(\mathbf{W}_{os/oe}^{r}(\mathbf{h} + \mathbf{s}_k) + \mathbf{b}_{os/oe}^{r}\big), \quad (3)$$

where $\mathbf{p}_{rk}^{os/oe}$ denotes the position probabilities under relation r w.r.t. the k-th subject, $(\mathbf{W}_{os}^{r}, \mathbf{b}_{os}^{r})$ and $(\mathbf{W}_{oe}^{r}, \mathbf{b}_{oe}^{r})$ are the classifiers' parameters of relation r. If the probabilities exceed some threshold, we set the corresponding tags to 1 otherwise 0. We heuristically set the threshold to 0.5 in our model. Then we match the nearest start-end position pair to identify subjects and objects. If an object o is identified under relation r w.r.t. a subject s, then (s, r, o) is extracted as a relational triple. We refer readers to (Wei et al., 2020) for more comprehensive descriptions of the extractor.

3.4 Training and Inference

We calculate a binary cross-entropy $f(\mathbf{y}, \mathbf{p}) = -\frac{1}{n}\sum_{i=1}^{n} y_i \log p_i + (1 - y_i)\log(1 - p_i)$ as the loss of a triple extractor's predictions:

$$\mathcal{L}_t = \sum_{*\in\{s,e\}} (f(\mathbf{y}^{s*}, \mathbf{p}^{s*}) + \sum_{r,k} f(\mathbf{y}_{rk}^{o*}, \mathbf{p}_{rk}^{o*})), \quad (4)$$

where $\mathbf{y}$ are the labels corresponding to the position probabilities $\mathbf{p}$. We apply a triple extractor to the encoder hidden states $\mathbf{h}^E$ to extract triples and obtain the encoder's triple loss, denoted as $\mathcal{L}_{\text{Enc}}$. Then we formulate the sentence reconstruction loss as a cross-entropy: $\mathcal{L}_{\text{LM}} = -\frac{1}{n}\sum_{i=1}^{n} \log p^{\text{LM}}(\hat{x}_i = x_i)$,where $\hat{x}_i$ is the i-th reconstructed token. However, we observe that training the decoder only using $\mathcal{L}_{\text{LM}}$ causes serious overfitting and hurts the performance. To reduce overfitting, we apply another extractor to the decoder hidden states $\mathbf{h}^D$ and compute the decoder's loss $\mathcal{L}_{\text{Dec}}$, which is equivalent to adding an auxiliary task for decoder training. Finally, we train our model with the joint loss $\mathcal{L} = \mathcal{L}_{\text{Enc}} + \mathcal{L}_{\text{LM}} + \mathcal{L}_{\text{Dec}}$. During inference, we only use the encoder's extracted triples because the decoder requires ground truth as its input.

4 Experiments

4.1 Datasets and Evaluation Metrics

We conduct our experiments on two widely used benchmark datasets: NYT (Riedel et al., 2010) and WebNLG (Gardent et al., 2017). NYT consists of sentences from the New York Times corpus and contains 24 relation types. WebNLG was proposed for natural language generation and used by Zeng et al. (2018) for relational triple extraction, which contains 171 relation types. Following Zeng et al. (2018), we split the sentences into three categories: *Normal*, *EntitypairOverlap* (EPO) and *SingleEntityOverlap* (SEO) according to different overlapping patterns of triples, as shown in Table 1. For a fair comparison, we employ the same partial match setting as various previous work (Wei et al., 2020; Chen et al., 2021) for evaluation. An extracted triple is regarded as correct only if the relation and the heads of both subject and object are all correct. We report the standard micro precision, recall, and F_1 scores on both datasets.

Dataset	NYT		WebNLG	
	Train	Test	Train	Test
Normal	37013	3266	1596	246
SEO	9782	1297	227	457
EPO	14735	978	3406	26
ALL	56195	5000	5019	703

Table 1: Statistics of NYT and WebNLG datasets.

4.2 Experimental Settings

We tune the hyper-parameters on the validation sets. We choose pre-trained checkpoints[2] of two T5 variants: T5_{BASE} and T5_{LARGE}, whose hidden dimensions are 768 and 1024, respectively. We adopt a 3-layer R-GCN and the hidden dimensions are 256. We apply the basis decomposition to regularize the R-GCN layers and the number of basis functions is 10. The MLP of R-GCN contains 2 layers and the hidden dimension is 128. We train our model using the Adam optimizer (Kingma and Ba, 2014) with the learning rate of $5e^{-4}$. We add 50% dropout (Srivastava et al., 2014) to all hidden layers of the R-GCN and the MLP. Following previous work (Chen et al., 2021), we set the max length of input sentences to 100. We train our model with

[2] https://huggingface.co/transformers/model_doc/t5v1.1.html

Method	# PLM Param.	NYT			WebNLG		
		Prec.	Rec.	F_1	Prec.	Rec.	F_1
NovelTagging (Zheng et al., 2017)	-	62.4	31.7	42.0	52.5	19.3	28.3
CopyRE (Zeng et al., 2018)	-	72.8	69.4	71.1	60.9	61.1	61.0
CASREL$_{BERT}$ (Wei et al., 2020)	110M	89.7	89.5	89.6	93.4	90.1	91.7
TPLinker$_{BERT}$ (Wang et al., 2020)	110M	91.3	92.5	91.9	91.8	92.0	91.9
SPN$_{BERT}$ (Sui et al., 2020)	110M	93.3	91.7	92.5	93.1	93.6	93.4
CGT$_{UniLM}$ (Ye et al., 2021)	110M	94.7	84.2	89.1	92.9	75.6	83.4
PFN$_{BERT}$ (Yan et al., 2021)	110M	-	-	92.4	-	-	93.6
TDEER$_{BERT}$ (Li et al., 2021)	110M	93.0	92.1	92.5	93.8	92.4	93.1
PRGC$_{BERT}$ (Zheng et al., 2021)	110M	93.3	91.9	92.6	94.0	92.1	93.0
‡R-BPtrNet$_{BERT}$ (Chen et al., 2021)	110M	92.7	92.5	92.6	93.7	92.8	93.3
‡R-BPtrNet$_{RoBERTa}$ (Chen et al., 2021)	355M	94.0	92.9	93.5	94.3	93.3	93.8
‡REBEL$_{BART}$ (Huguet et al., 2021)	406M	-	-	93.4	-	-	-
†CASREL$_{BERT}$	110M	89.3	90.1	89.7	92.8	90.9	91.8
†CASREL$_{T5\text{-}BASE\text{-}Encoder}$	110M	90.7	89.3	90.0	91.4	92.4	91.9
†CASREL$_{T5\text{-}BASE}$	220M	91.1	89.5	90.3	91.4	92.9	92.1
†MTG$_{T5\text{-}BASE}$	220M	94.9	92.4	93.7	94.6	93.3	93.9
†MTG$_{T5\text{-}LARGE}$	770M	**95.6**	**93.1**	**94.3**	**94.8**	**95.1**	**94.9**

Table 2: Performance of our MTG model and previous state-of-the-art models on the NYT and WebNLG test sets. The best scores are in bold and the second-best scores are underlined. † marks scores produced by our implementation of the CASREL extractor. ‡ marks models using entity type information.

the batch size of 40 on both datasets. To prevent overfitting, we stop the training process when the validation performance gains no improvement for 5 consecutive epochs. Then we load the parameters with the best validation performance, divide the learning rate by ten, and continue training for 20 epochs. Finally, we choose the best validation model and report scores on the test set.

4.3 Performance Evaluation

We report the evaluation results on the NYT and WebNLG test sets in Table 2. We compare our **MuTual Generation** model of **T**ext and **G**raph (**MTG**) with several state-of-the-art models: (1) **NovelTagging** (Zheng et al., 2017) proposed a novel sequence tagging scheme but ignored the overlapping triples. (2) **CopyRE** (Zeng et al., 2018) proposed to generate triple sequences with an end-to-end seq2seq model based on the copy mechanism. (3) **CASREL** (Wei et al., 2020) proposed a cascade binary tagging framework. (4) **TPLinker** (Wang et al., 2020) proposed a one-stage token pair linking model with a novel handshaking tagging scheme. (5) **SPN** (Sui et al., 2020) proposed to predict triple sets with a non-autoregressive decoder. (6) **CGT** (Ye et al., 2021) proposed a novel triple contrastive training object. (7) **PFN** (Yan et al., 2021) proposed a partition filter network to capture the interactions between entity and relation representations. (8) **TDEER** (Li et al., 2021) proposed a decoding schema that regards the relation as a translating operation from subject to objects. (9) **PRGC** (Zheng et al., 2021) proposed a potential relation and global correspondence model. (10) **R-BPtrNet** (Chen et al., 2021) proposed a reasoning pattern enhanced binary pointer network to extract implicit relational triples. (11) **REBEL** (Huguet Cabot and Navigli, 2021) proposed to generate linearized triples with an encoder-decoder language model.

From Table 2 we have several observations. First, our MTG$_{T5\text{-}BASE}$ model outperforms previous BERT-based models with similar amounts of PLM parameters for inference[3]. Also, it produces competitive performance to the models that incorporate entity type information and larger PLMs than T5$_{\text{BASE}}$. It indicates that our model effectively captures the relational reasoning patterns through

[3]During inference, we only use the MTG encoder's predictions involving half of the T5$_{\text{BASE}}$'s 220M parameters, which is similar to BERT$_{\text{BASE}}$'s 110M parameters.

Method	NYT								WebNLG							
	Nor.	SEO	EPO	N=1	N=2	N=3	N=4	N≥5	Nor.	SEO	EPO	N=1	N=2	N=3	N=4	N≥5
CopyRE	66.0	48.6	55.0	67.1	58.6	52.0	53.6	30.0	59.2	33.0	36.6	59.2	42.5	31.7	24.2	30.0
GraphRel	69.6	51.2	58.2	71.0	61.5	57.4	55.1	41.1	65.8	38.3	40.6	66.0	48.3	37.0	32.1	32.1
CASREL$_{BERT}$	87.3	91.4	92.0	88.2	90.3	91.9	94.2	83.7	89.4	92.2	94.7	89.3	90.8	94.2	92.4	90.9
TPLinker$_{BERT}$	90.1	93.4	94.0	90.0	92.9	93.1	96.1	90.0	87.9	92.5	95.3	88.0	90.1	94.6	93.3	91.6
SPN$_{\text{BERT}}$	90.8	94.0	94.1	90.9	93.4	94.2	95.5	90.6	-	-	-	-	-	-	-	-
PRGC$_{BERT}$	91.0	94.0	94.5	**91.1**	93.0	93.5	95.5	93.0	90.4	93.6	95.9	**89.9**	91.6	95.0	94.8	92.8
R-BPtrNet$_{BERT}$	90.4	94.4	95.2	89.5	93.1	93.5	96.7	91.3	89.5	93.9	96.1	88.5	91.4	96.2	94.9	94.2
R-BPtrNet$_{RoBERTa}$	91.2	95.3	96.1	90.5	93.6	94.2	97.7	92.1	89.9	94.4	97.4	89.3	91.7	96.5	95.8	94.8
MTG$_{T5\text{-}BASE}$	91.1	95.7	96.7	90.6	93.6	94.4	97.8	92.4	90.0	94.5	98.0	89.2	92.0	96.5	95.9	95.4
MTG$_{T5\text{-}LARGE}$	**91.3**	**96.2**	**97.9**	90.8	**94.7**	**96.4**	**98.4**	**93.2**	**90.7**	**95.6**	**98.7**	89.8	**92.4**	**97.8**	**97.3**	**96.5**

Table 3: F_1 scores on sentences with different overlapping patterns and different triple numbers. The best scores are in bold and the second-best scores are underlined. N stands for the number of triples in the sentence.

the mutual generation of text and graph and improves the performance. Second, MTG$_{T5\text{-}BASE}$ significantly outperforms CASREL$_{T5\text{-}BASE}$ and CASREL$_{T5\text{-}BASE\text{-}Encoder}$. We also notice that the two T5-based CASREL models perform only slightly better than CASREL$_{BERT}$. These results show that the improvements of our model come not primarily from the employment of T5, but from the mutual generation method we proposed. Finally, MTG$_{T5\text{-}LARGE}$ further outperforms MTG$_{T5\text{-}BASE}$ and other baseline methods. It indicates that the more powerful PLM brings more common-sense knowledge and conceptual facts to our model and helps capture the relation type inference logic more accurately.

4.4 Performance on Different Sentence Types

Following previous work (Wang et al., 2020; Chen et al., 2021), we split the test sets of the two datasets with the number of triples and the overlapping patterns to verify the ability of our model in handling complex sentences, as shown in Table 3. We observe that the MTG models bring significant improvements to the sentences with overlapping triples and with more than one triple. We argue that this is because these sentences have complicated interactions among their relational triples, which are more likely to require reasoning patterns to be extracted. Therefore, these sentences gain more improvements from our mutual generation model. In contrast, we observe that sentences without overlapping triples (and of course with only one triple) usually contain simple text patterns, thus receive

Method	Prec.	Rec.	F_1
MTG$_{T5\text{-}BASE}$	**94.9**	**92.4**	**93.7**
w/o R-GCN	93.4	91.5	92.5
w/o $\mathcal{L}_{\text{LM}}$	94.0	91.3	92.7
w/o $\mathcal{L}_{\text{Dec}}$	93.6	90.3	91.9
w/o All	90.7	89.3	90.0

Table 4: An ablation study of the MTG$_{T5\text{-}BASE}$ model.

Graph Edges	Prec.	Rec.	F_1
Full	**94.9**	**92.4**	**93.7**
Golden + Co-ref.	94.3	92.1	93.2
Golden + Reversed	94.5	92.1	93.3
Golden	93.8	91.9	92.8
None	93.4	91.5	92.5

Table 5: An ablation study of the graph edges.

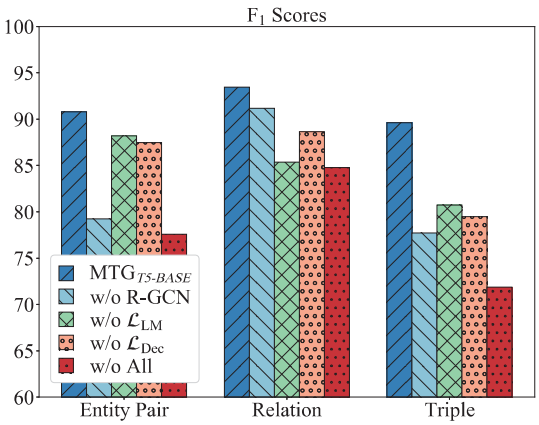

Figure 4: An ablation study on a manually selected subset with triples that require relational reasoning.

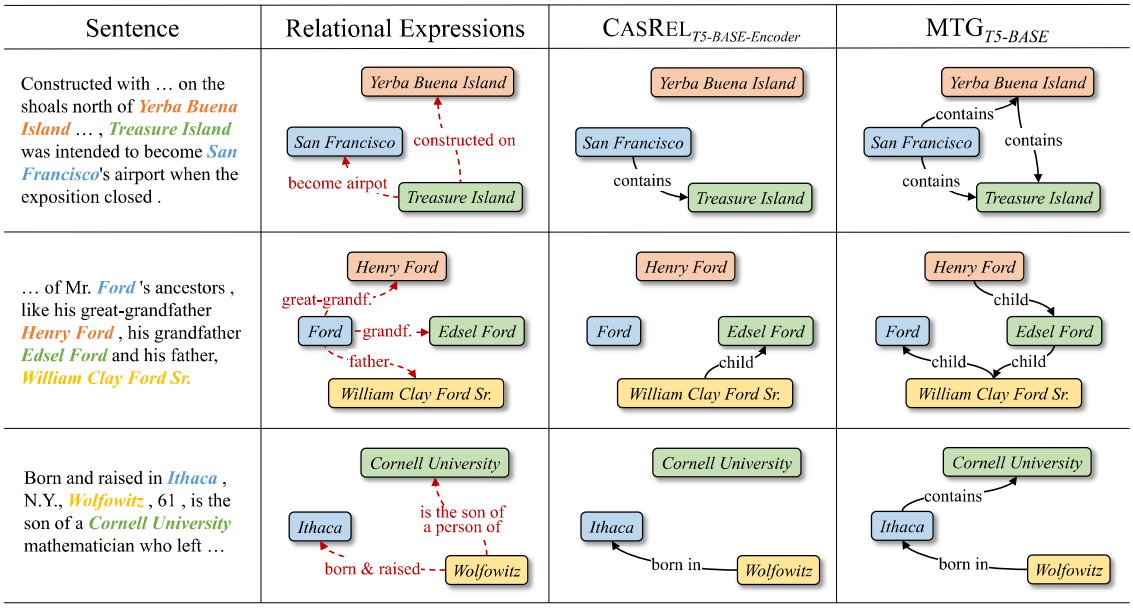

Figure 5: Examples of sentences with triples that require reasoning and the corresponding predictions from the MTG$_{T5\text{-}BASE}$ and CASREL$_{T5\text{-}BASE\text{-}Encoder}$ models. We distinguish different entities with different colors. Deep red dashed arrows indicate relational expressions of the sentence that helps extract and reason the triples.

limited benefit. Our model effectively learns relational reasoning patterns and improves the performance on complicated overlapping triples.

4.5 Ablation Study

To study the contribution of each component of our model, we run an ablation study on the NYT test set, as shown in Table 4. Note that when removing R-GCN, we average all node features and feed it into a fully-connected layer to obtain the graph representation $\mathbf{g}$. From Table 4 we observe that the R-GCN module and the sentence reconstruction task both have significant contributions to the model performance. The decoder's auxiliary loss also brings significant improvements because it prevents the model from overfitting to the sentence reconstruction task. Finally, the model without all three components (actually CASREL$_{T5\text{-}BASE\text{-}Encoder}$) produces the worst performance, which proves the effectiveness of our mutual generation method.

We also study the influence of three kinds of graph edges (Section 3.1), as shown in Table 5. We can observe that simply using the basic golden edges does not yield significant effects. Adding reversed golden edges and co-reference edges each bring more improvements to model performance because the flow of node information and the exploration of contextual information are more sufficient. Finally, the full graph yields the best performance, which demonstrates the effectiveness of our graph construction method.

To investigate the influence of each component of our model on relational reasoning, following Chen et al. (2021), we manually select 120 sentences with triples that need to be derived by relational reasoning and run the same ablation study on them. We illustrate the performance on the triples, entity pairs, and relation types in Figure 4. We can first observe that the R-GCN mainly contributes to the entity pair performance. It indicates the effectiveness of the text-to-graph generation in identifying potential multi-hop paths between the entities. Then, we observe that the sentence reconstruction mainly contributes to the performance of relation types, which shows the validity of the graph-to-text generation on capturing the type inference logic. The above observations demonstrate the effectiveness of our mutual generation method in learning relational reasoning patterns.

4.6 Case Study

Figure 5 shows the comparison of the MTG$_{T5\text{-}BASE}$ and CASREL$_{T5\text{-}BASE\text{-}Encoder}$ models on three example sentences. They have exactly the same model structures for inference and the only difference is that MTG$_{T5\text{-}BASE}$ is trained with our mutual generation method. In the first example, the including relation between "*San Francisco*" and "*Yerba Buena Island*" needs to be reasoned by understanding the geographical relationship of the three lo-

cations. The second example contains relational concepts "*great-grandfather*" and "*grandfather*", which indicate the parent-child relation chain of the persons. The third example implies that "*Cornell University*" is in "*Ithaca*" because a person of the university gives birth to a child in that place. We can observe that the CASREL model mainly concentrates on local text patterns, so it only extracts the superficial triples and even gets an error in the second example. Our MTG model effectively extracts the latent triples by capturing multi-hop interactions between entities and learning type inference logic from the relational expressions.

5 Conclusion

In this paper, we propose to learn relational reasoning patterns for relational triple extraction with mutual generation of text and graph. We construct a relational graph from the sentence and apply graph convolutions to identify multi-hop reasoning paths. We propose a sentence reconstruction task to explore the unlabeled conceptual expressions of the sentence for capturing the relation type inference logic along the paths. We conduct experiments on two benchmark datasets, and the results demonstrate the effectiveness of our method.

Acknowledgement

We would like to thank the anonymous reviewers for their constructive comments on this paper. This work was supported by the National Natural Science Foundation of China under Grant numbers U1936208, U1936216, and 61862002.

References

Gabor Angeli and Christopher D Manning. 2013. Philosophers are mortal: Inferring the truth of unseen facts. In *CoNLL*, pages 133–142.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *ACL-HLT*, pages 551–560, Portland, Oregon, USA. Association for Computational Linguistics.

Yubo Chen, Yunqi Zhang, Changran Hu, and Yongfeng Huang. 2021. Jointly extracting explicit and implicit relational triples with reasoning pattern enhanced binary pointer network. In *NAACL-HLT*, pages 5694–5703.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *KDD*, pages 601–610.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for nlg micro-planning. In *ACL*.

Matthew R Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *EMNLP*, pages 1774–1784.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. In *ACL*, pages 241–251.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *COLING*, pages 2537–2547.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. REBEL: Relation extraction by end-to-end language generation. In *Findings of EMNLP*, pages 2370–2381.

Ningning Jia, Xiang Cheng, and Sen Su. 2020. Improving knowledge graph embedding using locally and globally attentive relation paths. In *ECIR*, pages 17–32. Springer.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, pages 7871–7880.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *ACL*, pages 402–412.

Xianming Li, Xiaotian Luo, Chenghao Dong, Daichuan Yang, Beidi Luan, and Zhen He. 2021. Tdeer: An efficient translating decoding schema for joint extraction of entities and relations. In *EMNLP*, pages 8055–8064.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*, pages 1105–1116.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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. *JMLR*, 21:1–67.

Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *WWW*, pages 1015–1024.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*, pages 593–607. Springer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958.

Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, Xiangrong Zeng, and Shengping Liu. 2020. Joint entity and relation extraction with set prediction networks. *arXiv preprint arXiv:2011.01675*.

Bo-Hsiang Tseng, Jianpeng Cheng, Yimai Fang, and David Vandyke. 2020. A generative model for joint natural language understanding and generation. In *ACL*, pages 1795–1807.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *EMNLP-IJCNLP*, pages 5784–5789.

Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, and Limin Sun. 2020. Tplinker: Single-stage joint extraction of entities and relations through token pair linking. In *COLING*, pages 1572–1582.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *ACL*, pages 1476–1488, Online. Association for Computational Linguistics.

Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. 2021. A partition filter network for joint entity and relation extraction. In *EMNLP*, pages 185–197.

Hongbin Ye, Ningyu Zhang, Shumin Deng, Mosha Chen, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Contrastive triple extraction with generative transformer. In *AAAI*, volume 35, pages 14257–14265.

Xiaofeng Yu and Wai Lam. 2010. Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *COLING 2010: Posters*, pages 1399–1407.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *JMLR*, 3(Feb):1083–1106.

Daojian Zeng, Haoran Zhang, and Qianying Liu. 2020. Copymtl: Copy mechanism for joint extraction of entities and relations with multi-task learning. In *AAAI*, pages 9507–9514.

Xiangrong Zeng, Shizhu He, Daojian Zeng, Kang Liu, Shengping Liu, and Jun Zhao. 2019. Learning the extraction order of multiple relational facts in a sentence with reinforcement learning. In *EMNLP-IJCNLP*, pages 367–377.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *ACL*, pages 506–514.

Hengyi Zheng, Rui Wen, Xi Chen, Yifan Yang, Yunyan Zhang, Ziheng Zhang, Ningyu Zhang, Bin Qin, Xu Ming, and Yefeng Zheng. 2021. Prgc: Potential relation and global correspondence based joint relational triple extraction. In *ACL*, pages 6225–6235.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *ACL*, pages 1227–1236.

GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *ACL*, pages 427–434.

Hao Zhu, Yankai Lin, Zhiyuan Liu, Jie Fu, Tat-Seng Chua, and Maosong Sun. 2019. Graph neural networks with generated parameters for relation extraction. In *ACL*, pages 1331–1339.