Unsupervised Graph-Text Mutual Conversion with a Unified Pretrained Language Model

Yi Xu[1], Shuqian Sheng[1], Jiexing Qi[1], Luoyi Fu[1]*, Zhouhan Lin[1]
Xinbing Wang[1], Chenghu Zhou[2]
[1]Shanghai Jiao Tong University, Shanghai, China
[2]IGSNRR, Chinese Academy of Sciences, Beijing, China
{yixu98, susisheng, qi_jiexing, yiluofu, xwang8}@sjtu.edu.cn
lin.zhouhan@gmail.com

Abstract

Graph-to-text (G2T) generation and text-to-graph (T2G) triple extraction are two essential tasks for knowledge graphs. Existing unsupervised approaches become suitable candidates for jointly learning the two tasks due to their avoidance of using graph-text parallel data. However, they adopt multiple complex modules and still require entity information or relation type for training. To this end, we propose *INFINITY*, a simple yet effective unsupervised method with a unified pretrained language model that does not introduce external annotation tools or additional parallel information. It achieves fully unsupervised graph-text mutual conversion for the first time. Specifically, *INFINITY* treats both G2T and T2G as a bidirectional sequence generation task by fine-tuning only one pretrained seq2seq model. A novel back-translation-based framework is then designed to generate synthetic parallel data automatically. Besides, we investigate the impact of graph linearization and introduce the structure-aware fine-tuning strategy to alleviate possible performance deterioration via retaining structural information in graph sequences. As a fully unsupervised framework, *INFINITY* is empirically verified to outperform state-of-the-art baselines for G2T and T2G tasks. Additionally, we also devise a new training setting called cross learning for low-resource unsupervised information extraction.

1 Introduction

Graph-to-text (G2T) generation and text-to-graph (T2G) triple extraction are two mutually inverse tasks that are crucial to the domain of knowledge graphs (KGs). G2T verbalizes the structural information in KG with descriptive texts, which has attracted much attention to expand the application scope of KG, such as KG-based dialogue and Q&A system (Ji et al., 2022). As a primary task of information extraction, T2G aims to extract triples from text, the typical subtasks (He et al., 2020; Chen et al., 2022) of which include named entity recognition (NER) and relation extraction (RE). Figure 1 illustrates a training pair sample containing part of a knowledge graph and its corresponding text.

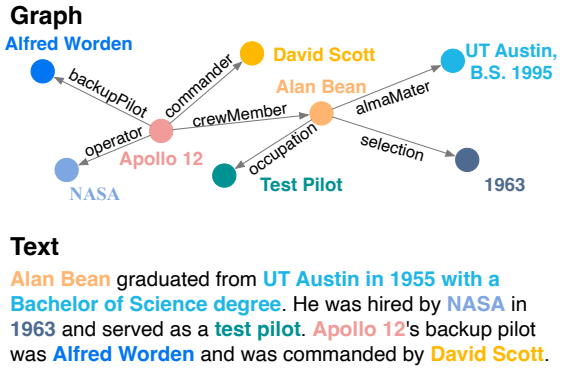

Figure 1: A pair of knowledge subgraph and its corresponding text.

G2T and T2G have been intensively studied respectively, mainly treated as two kinds of independent problems in a supervised way. Due to the success of pretrained language models (PLMs) (Raffel et al., 2020; Lewis et al., 2020), mainstream supervised methods have achieved considerable performance with fine-tuning or prompt learning paradigm (Ribeiro et al., 2021; Clive et al., 2021; Ye et al., 2021; Ke et al., 2021). However, these supervised methods require annotated data. Inspired by unsupervised machine translation approaches (Lample et al., 2018), recent work attempts to explore low-resource alternatives that avoid the requirement of graph-text pairs with unsupervised joint learning (Schmitt et al., 2020; Guo et al., 2020b). As illustrated in Figure 2, unsupervised methods consist of G2T modules and T2G modules with different parameters, which are trained jointly in an iterative manner through the two steps of back-translation: the generation step and training step. The outputs of the generation

* Luoyi Fu is the corresponding author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 5130–5144

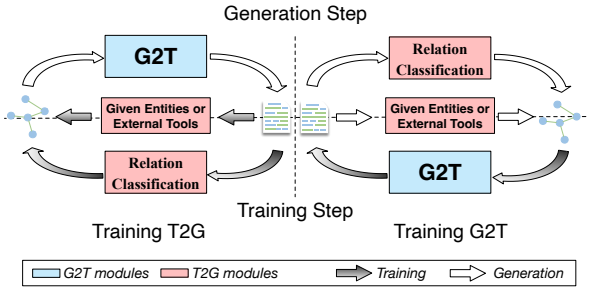

Figure 2: Framework of existing unsupervised models. The left part is the cycle of training T2G, and the right part is the cycle of training G2T.

step for the current modules serve as the supervised training signals for the other modules in the next iteration. Such an interactive and coupling process imperceptibly produces a lot of synthetic parallel data that is helpful to low-resource training. In this paper, we are thus motivated to focus on unsupervised learning of both G2T and T2G tasks in a joint framework.

As shown in Figure 2, unsupervised models share two major issues in order to be jointly trained. First, recent state-of-the-art models usually simplify the T2G task into relation classification with given entities (Jin et al., 2020). As a result, the text corpus has to seek external information extraction tools for the acquisition of entity annotations. Second, existing research branches on either G2T or T2G separately implement the two tasks using different neural modules, i.e., G2T modules and T2G modules, which contain numerous parameters that make it challenging to train and share information with each other (Schmitt et al., 2020).

To tackle the above issues, we design a novel back-translation-based framework called *INFINITY* that integrates G2T and T2G tasks under the unsupervised setting. Note that we name our framework as *INFINITY* since the overall architecture of the interaction between G2T and T2G resembles the shape of ∞ (Figure 3). We first investigate the power of seq2seq-based PLMs for G2T and T2G and propose to regard graph-text mutual conversion as two sequence generation tasks, where we manage to ensure the simultaneous generation of continuous synthetic pairs of graph-text sequences in a unified PLM-based module with the back-translation technique. In this way, *INFINITY* requires no additional neural networks beyond the PLM. Considering that linearizing graphs into sequences may cause possible performance deterioration, we equip *INFINITY* with structure-aware strategy. Specifically, we adopt the reward augmented maximum likelihood (Norouzi et al., 2016) for training losses to retain the order and structural information in the original dataset during the fine-tuning process. In contrast to prior unsupervised work (Schmitt et al., 2020; Guo et al., 2020b), *INFINITY* is entirely bootstrapped without the assistance from manual or automatic annotation tools.

We perform extensive experiments on two datasets: WebNLG (Gardent et al., 2017) and GenWiki (Jin et al., 2020), both of which belong to the very few benchmarks that can evaluate G2T and T2G jointly. The results show the superiority of *INFINITY* over existing methods. In addition, we also propose a newly designed training setting called cross learning, which makes it possible to train on large-scale datasets without parallel data. Thanks to its simplicity and efficiency, *INFINITY* can be quickly deployed on various scenarios for application. This work presents the following contributions:

- We are the first to take G2T and T2G as two unsupervised sequence generation tasks and propose *INFINITY*, a novel unsupervised framework for graph-text mutual conversion.
- *INFINITY* uses only **one** pretrained seq2seq model to generate synthetic parallel data iteratively and employs structure-aware fine-tuning strategy such as the reward augmented maximum likelihood to obtain structured graph sequences.
- *INFINITY* requires no parallel information or external annotation tools compared with other unsupervised models. With the help of cross learning, *INFINITY* is suitable for scenarios with large-scale datasets.
- We conduct extensive experiments to evaluate *INFINITY* on two benchmarks. The results demonstrate its superiority.

2 Related Work

2.1 Supervised Graph-text Models

As part of the data-to-text task, the key of G2T lies in capturing structural information and generating fluent texts. Some researchers (Koncel-Kedziorski et al., 2019; Li et al., 2021) design sophisticated architecture based on graph neural networks with heuristic rules to encode KGs. In addition, most

methods linearize the graph to sequence as input to models. However, graph linearization may lead to the loss of structural information. Past researches (Moryossef et al., 2019; Guo et al., 2020a; Frisoni et al., 2022) introduce different neural *planner* to determine the order of input triples before linearization. Recently, Ribeiro et al. (2021) investigate different PLMs for G2T generation. Clive et al. (2021) propose trainable control prefixes as prompts for PLM with finer-grained control during the process of text generation.

Regarding T2G, it aims to extract entities and relations (triples) from texts, which is a basic task in the domain of natural language processing and usually handled as a classification (tagging) problem to label roles for different tokens (Wei et al., 2020; Yan et al., 2021). Apart from these approaches, there emerge some triplet-generation models. For instance, CopyRE (Zeng et al., 2018) uses the idea of copy mechanism (Gu et al., 2016) for triple extraction. CPC (Ye et al., 2021) designs various positive and negative samples for direct graph sequence generation under a supervised setting.

2.2 Unsupervised Graph-text Models

As previously stated, due to the lack of parallel graph-text corpora, unsupervised models usually combine G2T and T2G into joint learning frameworks (Figure 2), which are motivated by unsupervised machine translation (Lample et al., 2018). There are only a few unsupervised graph-text models at present. Graph-Text Back Translator (GT-BT) (Schmitt et al., 2020) is the first approach to unsupervised text generation from KGs and can be used for semantic parsing simultaneously. CycleGT (Guo et al., 2020b) is another unsupervised training method that uses non-parallel graph and text data and iteratively back translates between the two forms. Although GT-BT and CycleGT employ back-translation for unsupervised settings, they simplify the T2G task to relation classification with given entities (Jin et al., 2020), which requires the text corpus to include entity annotations or equip with external information extraction tools. Therefore, these methods leak the information of parallel corpus in the training process to some extent.

3 Method

This section presents the proposed method *INFINITY*. We first define the tasks and notations. Then we describe the framework and training details in the following parts.

3.1 Formulation and Notations

Given two non-parallel datasets: a text corpus $\mathcal{T} = \{t_i\}_{i=1}^{N}$, and a graph dataset $\mathcal{G} = \{g_j\}_{j=1}^{M}$, where N and M are the numbers of text sequences and graphs, respectively. Each text sequence in $\mathcal{T}$ can be denoted as $t = (w_1, \cdots, w_L)$ with L tokens, where $w_i \in \mathcal{V}$ is the i-th token in t, and $\mathcal{V}$ is the vocabulary. Each graph in $\mathcal{G}$ consists of a set of triples, denoted as $g = \{(e_h, r, e_t) | e_h, e_t \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ and $\mathcal{R}$ represent the entity set and relation type set, respectively. Each entity $e \in \mathcal{E}$ is composed of several tokens formulated as $e = (w_1^e, \cdots, w_{L_e}^e), w_i^e \in \mathcal{V}$. Each relation type $r \in \mathcal{R}$ is also made up of tokens formulated as $r = (w_1^r, \cdots, w_{L_r}^r), w_i^r \in \mathcal{V}$. Similar to multilingual neural machine translation, we assume $\mathcal{T}$ and $\mathcal{G}$ share the same distribution of latent content z such as linguistic or semantic characteristics:

$$p(g) = \int_z p(g|z)p(z)dz, \tag{1}$$

$$p(t) = \int_z p(t|z)p(z)dz, \tag{2}$$

which is the key of unsupervised learning. In our unsupervised framework, both G2T and T2G are regarded as sequence generation tasks. G2T aims to generate a natural language text sequence from a knowledge subgraph, while T2G generates a triple sequence that represents the linearized graph where entities and relations exist in the given text. Since the graph itself is a set of triples, for a graph $g \in \mathcal{G}$, we adopt linearization strategy by concatenating all triples with special tokens $[H]$, $[R]$, $[T]$, and $[E]$ to specify the head entity, relation type, tail entity, and end of sequence respectively. The linearized graph is illustrated as follows:

$$\begin{aligned} &[H]\ e_h^1\ [R]\ r^1\ [T]\ e_t^1 \\ &[H]\ e_h^2\ [R]\ r^2\ [T]\ e_t^2 \\ &\qquad \cdots \\ &[H]\ e_h^{|g|}\ [R]\ r^{|g|}\ [T]\ e_t^{|g|}\ [E], \end{aligned} \tag{3}$$

where e_h^i, r^i, and e_t^i refer to the elements of the i-th triple in g. We can simply linearize the graph using the order of triples in the original dataset, and we will discuss how to keep structural information for graph linearization without designing sophisticated

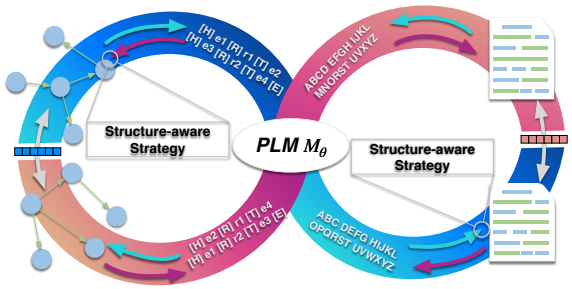

Figure 3: The overall architecture of *INFINITY*. The cycle of blue arrows illustrates the direction of G2T task while the cycle of magenta arrows illustrates the direction of T2G task.

methods that will lead to additional neural components. We only focus on the proposed framework rather than the inside neural components, although the linearization of the graph input is suboptimal.

3.2 Joint Training Framework of G2T & T2G

Different from unsupervised back-translation methods (Lample et al., 2018; Guo et al., 2020b), *INFINITY* only employs one neural network model, i.e., a single PLM. Therefore, the parameters of the framework are greatly reduced, while the PLM can observe both the original graphs and texts at the same time, which is easier for information sharing and model training. We denote the only PLM as M_θ, and the overall architecture of *INFINITY* is shown in Figure 3, which is shaped like ∞.

In Figure 3, the seq2seq-based PLM M_θ is in the centre of ∞. In the training process, the framework iteratively back-translates between graph dataset and text corpus. Here, the vocabulary embeddings are the same for G2T and T2G tasks, and the two tasks are both executed in M_θ. For simplicity and with a slight abuse of notation, we use the same symbol $M_\theta(\cdot)$ to represent the sequence generating function of the PLM, whether its output is discrete or continuous. The training process of *INFINITY* consists of two parts, the Graph $\rightarrow$ Text $\rightarrow$ Graph (GTG) cycle and the Text $\rightarrow$ Graph $\rightarrow$ Text (TGT) cycle. In each batch of the training process, M_θ is trained for the GTG cycle and the TGT cycle simultaneously. The training details are as follows.

Graph $\rightarrow$ Text $\rightarrow$ Graph. This cycle (the cycle of blue arrows) consists of two steps. In the first step, the goal is to generate a synthetic text sequence $\tilde{t}$. We first linearize the original graph into a triple sequence with special tokens. The linearized graph g is then fed to the encoder of M_θ, and the output $M_\theta(g)$ produced by the decoder of M_θ is the required intermediate result $\tilde{t}$. In the second step, M_θ further receives $\tilde{t}$ as an input text sequence and generates a back-translated graph. It is worth noting that $\tilde{t}$ is not a sequence of discrete tokens, but an embedding matrix of tokens, where each row of the matrix represents an embedding of a token. The PLM M_θ receives the synthetic text embedding and generates a back-translated graph $\tilde{g}$, which is used to align the original g through maximum likelihood estimation. Ideally, the back-translated graph should mimic the original graph g. Finally, parameters are updated with the guidance of estimation result.

Text $\rightarrow$ Graph $\rightarrow$ Text. Similarly, the other direction (the cycle of magenta arrows) also requires two steps. In the first step, a text sequence t is fed to PLM M_θ, and we denote the output synthetic graph sequence $M_\theta(t)$ as $\tilde{g}$. In the second step, $\tilde{g}$ is fed to M_θ. Here, $\tilde{g}$ is also in the embedding form. M_θ then generates a back-translated text $\tilde{t}$ based on the synthetic graph embedding, and parameters are trained on the basis of $\tilde{t}$ and the original t.

In summary, G2T and T2G can be optimized simultaneously in the proposed *INFINITY* with synthetic parallel pairs $(t, M_\theta(t))$ and $(g, M_\theta(g))$ i.e., $(t, \tilde{g})$ and $(g, \tilde{t})$. In both tasks, the model is expected to back-translate the synthetic result $M_\theta(t)$ and $M_\theta(g)$ into sequences that are roughly the same as the input t and g. The objective is as follows:

$$\begin{aligned}\mathcal{L} = \mathbb{E}_{g\in\mathcal{G}}[-\log P(g|M_\theta(g))]+ \\ \mathbb{E}_{t\in\mathcal{T}}[-\log P(t|M_\theta(t))]\end{aligned} \tag{4}$$

3.3 Structure-aware Fine-tuning

As a framework to solve the problem of bidirectional sequence generation, we need to consider how to retain more structural information in graphs as much as possible without introducing additional parameters. As already mentioned, recent approaches design sophisticated modules to order the input triples before graph linearization. However, these methods depend on supervision signals and make models more complex, which is not conducive to model generalization, let alone deployment to unsupervised settings.

From our perspective, graph linearization strategy hinders seq2seq-based PLM from capturing graph structure with maximum likelihood estimation (MLE) since MLE suffers from the exposure bias problem (Bengio et al., 2015). To this end, we can adopt reward augmented maximum likelihood

(RML) (Norouzi et al., 2016) which combines the primary form of MLE and the maximum reward expectation in reinforcement learning (RL). In this way, our training process is able to make rewards one of the training targets under the framework of MLE, which considers the structure of graphs and the order of texts. According to RML, the *exponentiated payoff distribution* connects MLE and RL objectives, and it can be easily incorporated into MLE-based training. In our framework, we define a distribution in the augmented space for graph dataset $\mathcal{G}$ as follows:

$$q(\tilde{g}|g;\tau) = \frac{exp(r(\tilde{g},g)/\tau)}{\sum_{\tilde{g}\in\tilde{\mathcal{G}}} exp(r(\tilde{g},g)/\tau)}, \quad (5)$$

where $\tilde{g} \in \tilde{\mathcal{G}}$ is the output hypothesis (possible generated sequence) of g, $r(\tilde{g}, g)$ denotes the reward function such as BLEU or F1 score, τ is the temperature to control the degree of regularization, and $\tau > 0$. Now, we modify the MLE-based objective function to:

$$\mathcal{L}^G_{RML} = \mathbb{E}_{g\in\mathcal{G}}[-\sum_{\tilde{g}\in\tilde{\mathcal{G}}} q(\tilde{g}|g;\tau)\log p(\tilde{g}|M_\theta(g))]. \quad (6)$$

In $\mathcal{L}^G_{RML}$, the predictive probability of the outputs in the original loss can be **smoothed** using their corresponding rewards with the distribution $q(\tilde{g}|g;\tau)$. For symmetry, RML can also be extended to our text corpus $\mathcal{T}$ similarly, which is shown as follows:

$$q(\tilde{t}|t;\tau) = \frac{exp(r(\tilde{t},t)/\tau)}{\sum_{\tilde{t}\in\tilde{\mathcal{T}}} exp(r(\tilde{t},t)/\tau)}, \quad (7)$$

$$\mathcal{L}^T_{RML} = \mathbb{E}_{t\in\mathcal{T}}[-\sum_{\tilde{t}\in\tilde{\mathcal{T}}} q(\tilde{t}|t;\tau)\log p(\tilde{t}|M_\theta(t))], \quad (8)$$

where $\tilde{t} \in \tilde{\mathcal{T}}$ is the output hypothesis of t. However, experiments show that the strategy will not significantly improve the performance when applied to text sequences.

The system of RML is simple and computationally efficient. One only needs to sample possible outputs $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{T}}$ from their corresponding exponentiated payoff distribution before training. Note that the structure-aware strategy is flexible in our framework (See Appendix for more details). In addition, existing unsupervised models **cannot employ RML** for graph extraction, which is defined as a relational classification problem rather than a sequence generation problem. In *INFINITY*, G2T and T2G tasks are jointly trained thanks to the shared parameters of M_θ. Compared with unsupervised machine translation, our method does not train a language model with the denoising objective on the two tasks due to the RML strategy. As a result, we optimize the loss function:

$$\mathcal{L} = \mathcal{L}^G_{RML} + \mathcal{L}^T_{RML}. \quad (9)$$

The detailed training process of unsupervised *INFINITY* is provided in Algorithm 1.

Algorithm 1 Training Unsupervised *INFINITY*

```
Initiate parameters of PLM M_θ^(1);
Obtain the distribution of q(g̃|g;τ) from G according to Equation 5;
Obtain the distribution of q(t̃|t;τ) from T according to Equation 7;
for i = 1 to N do
    G̃^(i) ← M_θ^(i)(M_θ^(i)(G));
    T̃^(i) ← M_θ^(i)(M_θ^(i)(T));
    Compute L^G_RML using G̃^(i) and G according to Equation 6;
    Compute L^T_RML using T̃^(i) and T according to Equation 8;
    L ← L^G_RML + L^T_RML;
    Fine-tune M_θ^(i) with L and obtain M_θ^(i+1);
end for
return M_θ^(N+1);
```

4 Experiments

This section conducts a series of experiments to evaluate the performance of *INFINITY*. We first introduce the datasets and baselines, then we provide the comparison results. Further, we implement extensive analytical experiments. At last, we show how cross learning works.

4.1 Datasets

Since our task is unsupervised, datasets with external information except for graphs and texts are not in our consideration. As a result, we select WebNLG (2017) (Gardent et al., 2017) and GenWiki (Jin et al., 2020) as our benchmarks, which

can evaluate G2T and T2G models at the same time. WebNLG is widely used in text generation and relation extraction, where each graph contains at most 7 triples. GenWiki is a new resource for unsupervised G2T generation, and we select two large domains (i.e., Sports and Games) of GenWiki. Tabel 1 presents the detailed statistics of these two datasets.

Dataset	Train	Valid	Test	Relation Types
WebNLG	13,036	1,642	4,928	373
GenWiki	48,020	1,000	10,000	250

Table 1: Statistics of benchmarks.

4.2 Baselines

4.2.1 Supervised Baselines

The intended application of *INFINITY* is in unsupervised scenarios. Thus, only relevant methods are considered. For G2T, we compare our model with a wide selection of PLM-free and PLM-based methods. PLM-free models include *StrongNeural*, *BestPlan* (Moryossef et al., 2019), *GraphWriter* (Koncel-Kedziorski et al., 2019), and *Planner* (Zhao et al., 2020), where *BestPlan* and *Planner* design different planners to order triples before linearization. PLM-based models include *T5-base* and *T5-large* (Ribeiro et al., 2021). As to T2G, we choose *OnePass* (Wang et al., 2019) and a state-of-the-art triple extraction model *CGT* (Ye et al., 2021) as our baselines. Moreover, we also implement a supervised version of *INFINITY* with aligned graph-text pairs, which serves as a reference for the upper bound of our unsupervised model. The supervised loss is: $\mathcal{L}^{sup} = \mathbb{E}_{(g,t)\in\mathcal{G}\times\mathcal{T}}[-\log P(g|t) - \log P(t|g)]$.

4.2.2 Unsupervised Baselines

Due to the limited research on unsupervised joint training, we selected almost all unsupervised models as baselines. *Rule-Based* (Schmitt et al., 2020) employs a heuristic algorithm to extract facts and concatenate text of each triplet. *Graph-Text Back Translator (GT-BT)* (Schmitt et al., 2020) adopts a series of denoising methods and applies a back-translation model with a POS tagger as external tool. *CycleGT* (Guo et al., 2020b) is derived from GT-BT, it jointly trains G2T and T2G tasks via cycle training, where the T2G is simplified to the relation classification task with given entities.

4.3 Training Settings and Evaluation Details

In our implementation, we use T5-base (Raffel et al., 2020) as the PLM for *INFINITY* since T5 is based on transformer and can handle multiple tasks well. We prepend graph prefix *Graph:* to the linearized graph sequence for G2T task and text prefix *Text:* for T2G task. In order to speed up the convergence of training, when generating synthetic intermediate outputs of texts, we discard embeddings of illegal tokens including $[H]$, $[R]$, $[T]$, and $[E]$ for the G2T task, which will not be fed to the encoder of the PLM in the following step. During the inference stage, we leverage the beam search to generate texts and linearized graphs. Additionally, for the T2G direction, we adopt the same heuristic rules recommended in prior work (Ye et al., 2021) to generate reasonable linearized graphs, where the special token $[R]$ (relation) should be followed by $[H]$ (head entity). The training and inference processes are carried out on NVIDIA GeForce RTX 3090.

We employ Adam as the optimizer. The beam size is set to 4 for both tasks. The learning rate is set to 1e-4. For G2T, we adopt several widely used automatic metrics, i.e., BLEU (Papineni et al., 2002), Meteor (Banerjee and Lavie, 2005), and CIDEr (Vedantam et al., 2015). BLEU and Meteor consider precision, recall, or F-score between generated and ground truth texts while CIDEr calculates the TF-IDF weights for each n-gram. For T2G, we use the micro F1 score to evaluate the quality of the generated triples. F1 results of entities and triples are both provided.

4.4 WebNLG Results

4.4.1 G2T Results

Table 2 presents the results of G2T task on the WebNLG dataset. For fairness, we report the results of *INFINITY* **without applying structure-aware strategy RML to texts**. It can be seen that our proposed method outperforms all other unsupervised baselines. The BLEU score of *INFINITY* is 2 points higher than CycleGT, we believe the independent G2T and T2G modules in CycleGT result in degraded performance. The results of *INFINITY* is even better than the level of some supervised models such as GraphWriter and Planner since they (the latter two) are PLM-free. Moreover, the performance of the supervised *INFINITY* is on par with T5-base and T5-large, and the supervised version can even deal with the T2G problem, which

can be attributed to the power of PLM and the joint optimization for the shared latent space.

	BLEU	METEOR	CIDEr
Supervised Models (G2T)			
StrongNeural	46.5	0.39	2.87
BestPlan	47.4	0.39	2.69
GraphWriter	45.8	0.36	3.14
Planner	52.9	0.45	3.72
T5-base	59.1	0.44	4.02
T5-large	59.3	0.44	4.03
Supervised *INFINITY*	58.8	0.44	3.99
Unsupervised Models (Given Entities / External Tools)			
Rule-Based	18.3	0.34	-
GT-BT	37.7	0.36	-
CycleGT	55.5	0.44	3.81
Unsupervised Models			
INFINITY	58.0	0.44	3.89

Table 2: G2T performances of different models on WebNLG dataset. CIDEr results and corresponding codes are not provided in Rule-Based and GT-BT.

4.4.2 T2G Results

For the T2G task, it should be mentioned that the three compared unsupervised models RuleBased, GT-BT, and CycleGT, are given entities as a relation classification task, so they have a 100% F1 score of entities naturally and cannot employ RML loss for graph sequences. As can be seen from Table 3, our model's F1 (triple) score is 61.7, which is superior to all other unsupervised models under the circumstance that all entities are unknown. Rule-Based model cannot extract any triples. The supervised *INFINITY* shows better results than the unsupervised one in terms of entity recognition, whereas its performance is inferior to other supervised methods since our model only uses the T5-base PLM and does not equip other sophisticated modules.

	F1 (entity)	F1 (triple)
Supervised Models (T2G)		
OnePass	N/A	66.2
CGT	N/A	83.4
Supervised *INFINITY*	95.0	59.3
Unsupervised Models (Given Entities / External Tools)		
Rule-Based	100.0	0.0
GT-BT	100.0	39.1
CycleGT	100.0	58.4
Unsupervised Models		
INFINITY	93.9	61.7

Table 3: T2G performances of different models on WebNLG dataset. N/A means the model is not applicable to extract entities.

4.5 GenWiki Results

Unlike the WebNLG dataset, GenWiki is specially collected for unsupervised G2T task, where graph elements do not necessarily exist in the text. Moreover, the entities extracted from the text are also not necessarily contained in the ground truth graph, which makes it challenging to generate informative outputs. Hence, some supervised baselines are not applicable to this dataset. Since the codes of Rule-Based and GT-BT (Schmitt et al., 2020) are not provided, we use our implemented Rule-Based model as the baseline. In Table 4, our proposed method shows better results than GraphWriter and Rule-Based model, but the BLEU value of *INFINITY* is lower than CycleGT. The reason is that CycleGT has known all tokens of entities and relation types for T2G task, which can be used as external information to achieve better performance during the training process. As a result, *INFINITY* can only generate the tokens of entities and relations that appear in the original texts. In other words, our model may substitute the ground truth tokens with other words but remain the similar meanings. For example, the original relation *birthYear* may be predicted as *birthDay* in *INFINITY*.

	G2T		T2G
	BLEU	CIDEr	F1 (triple)
Supervised Models			
GraphWriter	29.7	2.68	N/A
T5-base	45.7	3.74	N/A
T5-large	47.1	3.74	N/A
Supervised *INFINITY*	43.6	3.44	33.8
Unsupervised Models (Given Entities / External Tools)			
Rule-Based (our implementation)	13.9	1.26	0.0
CycleGT	38.5	3.50	34.2
Unsupervised Models			
INFINITY	34.3	2.50	23.4

Table 4: G2T and T2G performances of different models on GenWiki dataset.

	G2T		T2G
	BLEU	CIDEr	F1 (triple)
Supervised *INFINITY*	**58.8**	**3.99**	59.3
w/o RML	54.3	3.58	51.5
w. RML for text & graph	57.3	3.89	59.7
w. RML for text	56.2	3.67	53.8
w. RML for graph (ours)	58.0	3.89	**61.7**

Table 5: Ablation analysis on WebNLG dataset. The version with RML for graph is used as our reported results.

	WebNLG				GenWiki			
	G2T		T2G		G2T		T2G	
	BLEU	CIDEr	F1 (entity)	F1 (triple)	BLEU	CIDEr	F1 (entity)	F1 (triple)
WebNLG.G × GenWiki.T	34.8	2.04	89.1	45.2	21.6	1.41	59.2	1.2
WebNLG.T × GenWiki.G	45.6	2.82	91.9	19.5	16.1	1.13	65.6	9.1
WebNLG	58.0	3.89	93.9	61.7	N/A	N/A	N/A	N/A
GenWiki	N/A	N/A	N/A	N/A	34.3	2.50	97.0	23.4

Table 6: Analysis of cross learning on WebNLG and GenWiki datasets. $dataset.G$ means the graph data in $dataset$ and $dataset.T$ denotes the text corpus in $dataset$. The last two rows are the results of training with the graphs and texts on a single dataset, where they are only evaluated on their corresponding benchmark.

4.6 Detailed Analysis

4.6.1 Ablation Study

We use the WebNLG dataset for ablation analysis. As shown in Table 5, the supervised *INFINITY* shows the best results on the G2T task. The performance of *INFINITY* without reward augmented losses (w/o RML) is worse than any other versions, especially for T2G task, but it can still compete with other unsupervised models such as cycleGT. Applying structure-aware strategy to both text and graph makes the model capture more order and structural information in the datasets, and it obtains significant improvement. We also evaluate variants that only adopt one side reward augmented loss. *INFINITY* with RML for graph demonstrates the best performance except for the supervised one. This is because the PLM itself performs well on texts, and the improvement of RML for text is limited. Therefore, we use the version with RML for graph as our final reported model.

4.6.2 Analysis of Input Scale

We investigate how the performance of *INFINITY* scales with the amount of input data since our method models graph-text conversion as two sequence generation tasks. For G2T, we divide the input graphs into groups based on the number of triples and calculate the mean BLEU value of each group of text sequences obtained by *INFINITY*. Similarly, as to T2G, we group the input data according to the length of the text sequences and count the F1 value of the output graphs. Figure 4 shows the results of different input sizes on the WebNLG dataset. It can be observed that the BLEU values decrease with the increase of the number of triples (≤ 5). When the number of triples exceeds 5, the BLEU values abnormally increase because there are fewer samples of long triples (about 3%). Besides, the F1 value is insensitive to the input size of texts. One of the possible explanations is that $\mathcal{T}$ and $\mathcal{G}$ share the same distribution of latent content, and *INFINITY* only makes minor modifications to the text sequence for transformation. Further, *INFINITY* can be applied to datasets with larger and more complex graphs. The performance depends on how much latent content graph and text sequences share. Also, memory requirements quadruple when doubling the input length due to the adopted PLM limitation (Raffel et al., 2020). In practice, we suggest using community detection algorithms to divide a large graph into multiple subgraphs for training.

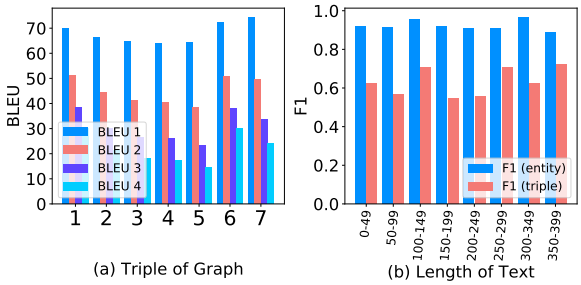

Figure 4: Results of different input sizes on WebNLG dataset.

4.7 Cross Learning

As mentioned in Section 3.1, we assume $\mathcal{T}$ and $\mathcal{G}$ share the same latent content. In the same dataset, $\mathcal{T}$ and $\mathcal{G}$ have the same domain knowledge, whereas different datasets can only share the language. In the latter case, to analyze the scalability of *INFINITY*, we propose a new training setting called cross learning, where we only use the graph (or text) data of WebNLG and text (or graph) corpus of GenWiki for training. Table 6 shows the results, where $dataset.G$ means the graph in $dataset$ while $dataset.T$ denotes the text in $dataset$. We can see *INFINITY* works well under the setting of cross learning, which cannot be accomplished by other unsupervised models such as CycleGT

since they require entities and relation types for both tasks. However, the T2G performance of GenWiki is worse than WebNLG because the tokens of relations and texts rarely overlap in GenWiki. In summary, *INFINITY* provides a low-resource approach to deploy PLM on large-scale unannotated datasets for application. For example, in the absence of a corresponding graph corpus, we can use public knowledge graphs datasets to train *INFINITY* model so as to extract graph triples from any given English literature.

5 Conclusion and Future Work

In this manuscript, we propose *INFINITY*, a simple unsupervised approach to graph-text mutual conversion. The key idea of *INFINITY* is to utilize one seq2seq-based PLM to convert graphs and texts from each other with the framework of back-translation. Unlike existing unsupervised methods, our model requires no additional external information or tools beyond the non-parallel graph and text corpus, so it is easy to be quickly deployed to industrial scenarios. Experiments show that *INFINITY* achieves promising results compared to state-of-the-art baselines. For future work, we plan to explore the capability of prompt learning by appealing to precise controls over different attention layers in PLMs.

Limitations

One limitation of the proposed method, INFINITY, is that it is currently limited to fully unsupervised and not incorporating any parallel data which may lead to performance deterioration in uncommon scenarios. Furthermore, it only works well with languages having limited morphology such as English and may not perform as well on languages with complex morphology. Finally, the method may have low scalability to long text as it requires large GPU resources. These limitations inspire further investigation to improve the performance and applicability of the method.

Acknowledgements

This work was supported by NSF China (No.42050105, 62020106005, 62061146002, 61960206002), Shanghai Pilot Program for Basic Research - Shanghai Jiao Tong University.

References

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, Ann Arbor, Michigan. Association for Computational Linguistics.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1171–1179, Cambridge, MA, USA. MIT Press.

Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for amr parsing. 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 3799–3809.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 2778–2788, New York, NY, USA. Association for Computing Machinery.

Jordan Clive, Kris Cao, and Marek Rei. 2021. Control prefixes for text generation. *arXiv preprint arXiv:2110.08329*.

Giacomo Frisoni, Gianluca Moro, and Lorenzo Balzani. 2022. Text-to-text extraction and verbalization of biomedical event graphs. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2692–2710, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Qipeng Guo, Zhijing Jin, Ning Dai, Xipeng Qiu, Xiangyang Xue, David Wipf, and Zheng Zhang. 2020a.

$\sqrt{}^{2}$: A plan-and-pretrain approach for knowledge graph-to-text generation. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 100–106, Dublin, Ireland (Virtual). Association for Computational Linguistics.

Qipeng Guo, Zhijing Jin, Xipeng Qiu, Weinan Zhang, David Wipf, and Zheng Zhang. 2020b. CycleGT: Unsupervised graph-to-text and text-to-graph generation via cycle training. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 77–88, Dublin, Ireland (Virtual). Association for Computational Linguistics.

Qizhen He, Liang Wu, Yida Yin, and Heming Cai. 2020. Knowledge-graph augmented word representations for named entity recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:7919–7926.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494—514.

Zhijing Jin, Qipeng Guo, Xipeng Qiu, and Zheng Zhang. 2020. GenWiki: A dataset of 1.3 million content-sharing text and graphs for unsupervised graph-to-text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Pei Ke, Haozhe Ji, Yu Ran, Xin Cui, Liwei Wang, Linfeng Song, Xiaoyan Zhu, and Minlie Huang. 2021. JointGT: Graph-text joint representation learning for text generation from knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2526–2538, Online. Association for Computational Linguistics.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2284–2293, Minneapolis, Minnesota. Association for Computational Linguistics.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5039–5049, Brussels, Belgium. Association for Computational Linguistics.

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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2021. Few-shot knowledge graph-to-text generation with pre-trained language models. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1558–1568, Online. Association for Computational Linguistics.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277, Minneapolis, Minnesota. Association for Computational Linguistics.

Mohammad Norouzi, Samy Bengio, zhifeng Chen, Navdeep Jaitly, Mike Schuster, Yonghui Wu, and Dale Schuurmans. 2016. Reward augmented maximum likelihood for neural structured prediction. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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(140):1–67.

Leonardo F. R. Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2021. Investigating pretrained language models for graph-to-text generation. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, pages 211–227, Online. Association for Computational Linguistics.

Martin Schmitt, Sahand Sharifzadeh, Volker Tresp, and Hinrich Schütze. 2020. An unsupervised joint system for text generation from knowledge graphs and semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7117–7130, Online. Association for Computational Linguistics.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *CVPR*.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019. Extracting multiple-relations in one-pass with pre-trained transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1371–1377, Florence, Italy. Association for Computational Linguistics.

Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 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 *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 185–197, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hongbin Ye, Ningyu Zhang, Shumin Deng, Mosha Chen, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Contrastive triple extraction with generative transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14257–14265.

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 *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 506–514, Melbourne, Australia. Association for Computational Linguistics.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2481–2491, Online. Association for Computational Linguistics.

A RML Sampling

As mentioned earlier, one only needs to sample possible outputs $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{T}}$ from their corresponding exponentiated payoff distribution before training. According to Norouzi et al. (2016), it is difficult to sample with BLEU or F1 score since the distribution is intractable to compute. Thus, we utilize the importance sampling method with the distribution of hamming distance between the original sequence and its hypothesis. For the graph dataset, L sequences of linearized graphs associated with g are sampled from a tractable proposal distribution $q(\tilde{g}|g;\tau)$, i.e., hamming distance sampling. Then, we obtain the importance weight ω_l of each sampled sequence $\tilde{g}^l$:

$$\omega_l \approx \frac{exp(r(\tilde{g}^l, g)/\tau)/q(\tilde{g}^l|g;\tau)}{\sum_{k=1}^{L} exp(r(\tilde{g}^k, g)/\tau)/q(\tilde{g}^k|g;\tau)}, \quad (10)$$

which replaces the proposal distribution by reweighing the samples in loss $\mathcal{L}^G_{RML}$. More details about importance sampling can be found in Norouzi et al. (2016).

B Common and Heuristic Structure-aware Strategies

The structure-aware strategy is flexible in *INFINITY*, we can adopt other structure-aware strategies.

Generally, PLMs are pretrained based on a large number of text corpora, and they are not fed with linearized graphs. Thus, the following denoising loss with perturbed linearized graph structure is a suitable way to warm up PLMs:

$$\mathcal{L}^G_{lm} = \mathbb{E}_{g\in\mathcal{G}}[-\log p(g|corrupt(g);\theta)], \quad (11)$$

where $corrupt(\cdot)$ is a function that generates corrupted graph sequence. Possible operations on the graph structure include but are not limited to *replacement, discarding, duplication, and swapping*. By this means, PLM is capable of remembering the structure of graphs.

Besides, BFS and DFS traversal-based graph linearization are another two common methods adopted by many supervised generation models (Li et al., 2021; Cai and Lam, 2019), especially in the field of semantic role labeling and AMR semantic parsing. In our unsupervised setting, we can execute BFS or DFS from the node with the maximum degree value to obtain a reasonable graph sequence, where nodes with similar semantics are close.

In the experiments, we find that RML can achieve the best performance, thus we adopt RML as the only strategy in *INFINITY*.

C Case Study and Error Analysis

To analyze the generation performance and drawbacks of *INFINITY*, we **select two representative instances** shown in Table 7, where the ground truth and generated sequences are provided. As to the first case, the generated text is consistent with the ground truth, with only a few slight differences, and the generated triples are exactly the same as the real ones. The second instance contains two sentences and five triples, which has several typical errors. The order of the generated text is inconsistent with the original text, and there are some semantic errors. The generated triples are all reasonable, but the first fact with the relation *related Mean Of Transportation* is missing. The boundary of the last generated triple is wrong, where $[R]$ does not appear in the proper position, and $[T]$ is missing. It is still challenging to capture fine-grained entity or relation boundaries without supervised information such as relation type.

	Ground Truth Sequence	Generated Sequence	Main Error
Text 1	Arlington in Texas is located at 184.0 metres above sea level and has a total area of 258.2 square kilometres.	Arlington , Texas is 184.0 above sea level and has a total area of 258.2 square kilometres.	Rephrasing
Graph 1	[H] Arlington Texas [R] elevation Above The Sea Level [T] 184.0	[H] Arlington Texas [R] elevation Above The Sea Level [T] 184.0	Exact Match
	[H] Arlington Texas [R] area Total [T] 258.2 square kilometres [E]	[H] Arlington Texas [R] area Total [T] 258.2 square kilometres [E]	
Text 2	The Aston Martin V8, manufactured by Aston Martin, is related to the Aston Martin DBS and was succeeded by the Aston Martin Vantage. Its engine volume is 5.3 litres. and it is assembled at Newport Pagnell.	The Aston Martin V8, with a 5.3 litre engine, is a related transport vehicle to the Aston Martin DBS. It is the successor to the Newport Pagnell Aston Martin Vantage.	Senetence Missing
Graph 2	[H] Aston Martin V8 [R] related Mean Of Transportation [T] Aston Martin DBS	[H] Aston Martin V8 [R] manufacturer [T] Aston Martin	
	[H] Aston Martin DBS [R] successor [T] Aston Martin Vantage	[H] Aston Martin DBS [R] succeeded By [T] Aston Martin Vantage	Triple Missing
	[H] Aston Martin V8 [R] engine [T] 5.3 litres	[H] AstonMartin V8 [R] engine Volume [T] 5.3 litres	Wrong Boundary
	[H] Aston Martin V8 [R] assembly [T] Newport Pagnell	[H] Aston Martin assembly location [R] Newport Pagnell [E]	Rephrasing
	[H] Aston Martin V8 [R] manufacturer [T] Aston Martin [E]		

Table 7: Case study and error analysis on two selected representative instances. The blue background represents nuances that do not change semantic meanings, while the orange background represents information loss. For graphs, we arrange the triples in the order they are in the ground truth or generated sequences.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Appendix D

☑ A2. Did you discuss any potential risks of your work?
Appendix D

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

4.1

☑ B1. Did you cite the creators of artifacts you used?
4.1

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
It is widely used, the license is contained in their original package.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
4.1

C ☑ Did you run computational experiments?

4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4.3 and Appendix C

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
4.3 and Appendix C

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4.4 and 4.5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
4.3 and Appendix C

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.