Natural Language Generation from Large-Scale Open-Domain Knowledge Graphs

Anonymous ACL submission

Abstract

In generating natural language descriptions for knowledge graph triples, the effectiveness of graph-to-text models being trained on either small-scale, human-annotated datasets or large-scale but monotonously shaped graphs remain largely not refined or assessed for realistic large-scale, open-domain settings. This paper presents novel methods and a new dataset that seeks to fill this gap between graph-to-text models and real-world knowledge graphs. The dataset consists of input graph-output text pairs formed by aligning Wikipedia sentences with Freebase. With regard to the methods, fine-tuning transformer-based pre-trained language models (PLMs) has achieved state-of-the-art performance, however, this method suffers from information hallucination—the generated text may contain fabricated facts that are not present in input graphs. To tackle this challenge, we propose a novel approach that trims the sentence to remove its parts that are not present in the graph, based on analyzing the sentence's dependency parsing tree. Experiment results and human evaluation show that this approach helps reduce hallucination for models fine-tuned using both our dataset and other existing datasets. The dataset, source code, and fine-tuned models for this project can be found at [1].

1 Introduction

The task of *graph-to-text generation* aims to automatically generate natural language descriptions of knowledge graphs (KGs). A knowledge graph stores factual information as triples in the form of (subject, predicate, object), e.g., (Ludvig van Beethoven, *profession*, Composer). The graph-to-text generation task can be accomplished by constructing machine learning models (Clive et al., 2021; Castro Ferreira et al., 2019; Trisedya et al., 2018; Marcheggiani and Perez-Beltrachini, 2018; Gardent et al., 2017b). The input to such a model is a graph itself—a small fragment of triples from a knowledge graph, incoming from the outcome of some upstream operation, such as search, query, and data mining. The output is a textual sequence that describes the fragment of triples.

In graph-to-text generation, the preciseness and naturalness of the textual narration of graph fragments are of utmost importance. Attaining high quality in this regard can be particularly challenging when the knowledge graphs are *large-scale* and *open-domain*. Specifically, widely used benchmark datasets in this line of research either are *hand-crafted* and *monotonous*, e.g., WebNLG (Gardent et al., 2017a), or their input fragments being narrated are of *simple, special* formations only, e.g., EventNarrative (Colas et al., 2021) and TEKGEN Corpus (Agarwal et al., 2021). On the contrary, 1) it is impractical to curate such a hand-crafted dataset for knowledge graphs with many different types of entities and relations from a large number of topic domains. 2) The natural language humans use to describe graph fragments ought to be diverse instead of scripted to adhere to monotonous templates. 3) In practical scenarios the input fragments being narrated can be of *complex, general* rather than simple, special formations. The effectiveness of existing graph-to-text models, being trained on the datasets as mentioned earlier, thus remains largely *not* assessed for realistic large-scale settings.

This paper presents novel methods and a new dataset that aims to fill the aforementioned gap between graph-to-text models and real-world knowledge graphs. The ensuing discussion in the rest of this section further articulates these challenges and our solutions.

First, graph-to-text models are trained using pairs of input graph fragments and output textual descriptions. Most prior models were trained on small hand-crafted datasets that contain limited en-

[1] https://anonymous.4open.science/r/graphnarrator-40F6/

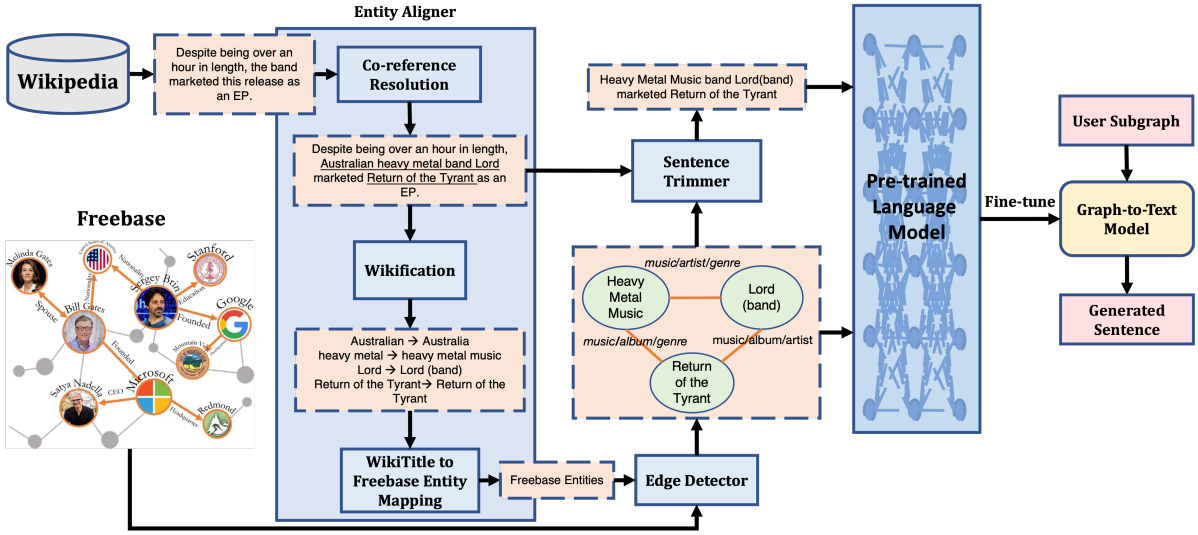

Figure 1: Overview of our framework

tity types and relations. In contrast, large-scale, open-domain knowledge graphs have millions of entities and hundreds of thousands of relations. The handcrafted approach cannot scale to these real-world datasets, as it is impossible to manually write training graph-text pairs for such a vast number of different entity types and relations. Instead, this paper presents a method to *automatically* generate training examples by aligning Wikipedia texts with Freebase.

Second, another limitation of hand-crafted datasets such as WebNLG is they are overly *clean*, in the sense that text descriptions in its training examples tend to follow monotonous templates, likely because the examples were written by a small number of human contributors. This limits the capacity of trained models to narrate graph fragments in diverse ways. In contrast, with our approach of automatically extracting training graph-text pairs via Freebase-Wikipedia alignment, the training examples allow a model to learn from many Wikipedia authors' diverse narrations of each type of relation.

Third, the graph fragments in existing datasets are largely limited to simple *star graphs*. Each star graph consists of a central entity and several of its one-hop neighbors. The hand-crafted WebNLG includes star graphs in the majority. Automatically generated datasets EventNarrative (Colas et al., 2021) and TEKGEN Corpus (Agarwal et al., 2021) collected graph-text pairs by aligning Wikipedia sentences with Wikidata, similar to how we aligned Wikipedia with Freebase. But, given their particular way of alignment, their datasets are dominated by star graphs, which constitute 94% and 96% of the graphs in EventNarrative and TEKGEN Corpus, respectively. Another automatically-collected dataset, AGENDA (Koncel-Kedziorski et al., 2019), has only 2% star graphs. But it only contains 7 distinct relations in the domain of scientific research. On the contrary, the training examples in our new dataset include general graphs beyond simple star graphs. This key difference makes our dataset and methods more generally applicable, as fragments to be narrated in real applications are more complex and general than star graphs. They could take various formations—paths, triangles, diamonds, and so on.

Given the efficacy of fine-tuning PLMs in producing graph-to-text models, we adopt the same approach. As pointed out in (Agarwal et al., 2021; Dušek et al., 2018), though, this approach may suffer from information *hallucination*, i.e., the resulting model may fabricate facts in output texts that are not present in input graphs. For example, given a two-triple input graph ((`Neff Maiava`, *date of birth*, `01 May 1924`), (`Neff Maiava`, *date of death*, `21 April 2018`)), (Agarwal et al., 2021) reported their model generates *"Neff Maiava (1 May 1924 - 21 April 2018) was an Albanian actor"* in which Maiava's profession and citizenship were not mentioned in the input graph, in fact, `Neff Maiavs` was an American wrestler. Fabrications in training data are the culprit of hallucination (Agarwal et al., 2021). More specifically, the reason is the training data contain textual descriptions with information not found in input graphs. This is evidenced by the fact that,

while hallucination is frequent on the TEKGEN Corpus used in (Agarwal et al., 2021), it is seldom observed on WebNLG. The clean, manually-crafted WebNLG seldom has hallucinated facts in its training data. In contrast, automatically extracted graph-text pairs in TEKGEN contain more hallucinated facts due to errors in extracting entities and relationships.

Very few have studied how to mitigate hallucination, except for (Agarwal et al., 2021; Wang et al., 2021) which further fine-tuned PLMs on WebNLG after fine-tuning on noisy automatically-extracted datasets. We take a different approach. Given a graph-text pair in our automatically extracted training data, we trim the text, i.e., a Wikipedia sentence, to remove parts not reflected in the extracted graph. The trimming is guided by the shortest paths between extracted entity pairs in the sentence's dependency parsing tree (Kübler et al., 2009). We conducted automatic and human evaluations of descriptions generated by fine-tuning PLMs (BART and T5), using original Wikipedia sentences and trimmed sentences as target outputs, respectively. The results show fewer hallucinations in descriptions using trimmed sentences.

The main contributions of this paper are as follows:

- A novel approach to enhance the training data for natural language generation from knowledge graphs, allowing PLMs to better handle diverse graph structures and improve real-world applications.
- The introduction of the "Sentence Trimmer" method to address the problem of "hallucination" in PLMs when fine-tuning for graph-to-text tasks on large-scale, open-domain knowledge graphs, resulting in more credible outputs.
- The release of a large-scale graph-to-text dataset, built using Freebase as the knowledge graph and trimmed Wikipedia sentences as text descriptions, with a more diverse range of graph structures compared to existing datasets.

2 Related Work

The models in previous studies of graph-to-text often use an encoder-decoder structure (Sutskever et al., 2014), where the encoder learns representations of input graphs and the decoder translates such representations into token sequences describing the input graphs. Based on the way they represent graphs, these models can be categorized into two classes. In one class there are sequence-to-sequence models that encode linearized graphs' token sequences with LSTMs (Trisedya et al., 2018; Gardent et al., 2017b) or Transformers (Castro Ferreira et al., 2019). In the other class, there are models that use a graph encoder to capture the structure of knowledge graphs (Schmitt et al., 2021; Ribeiro et al., 2020; Marcheggiani and Perez-Beltrachini, 2018). The state-of-the-art results on WebNLG are from some models (Ribeiro et al., 2021; Wang et al., 2021; Clive et al., 2021) in the aforementioned first class. Specifically, these models fine-tune transformer-based PLMs such as GPT-2 (Radford et al., 2019), BART (Lewis et al., 2020), and T5 (Raffel et al., 2020). Their state-of-the-art results demonstrate PLMs' generation power for graph-to-text tasks.

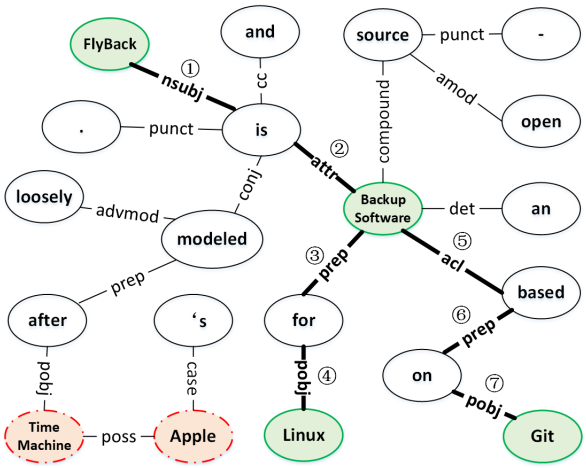

Figure 2: Dependency parse tree of the sentence "FlyBack is an open-source Backup Software for Linux based on Git and modeled loosely after Apple's Time Machine."

3 Methods

A knowledge graph, denoted as $\mathcal{G}$, is composed of a set of triples (s, p, o), where the subject s and object o are entities from the entity space E, and predicate p are from the relation space R. The task of generating text from a subgraph $G \subset \mathcal{G}$ can be formulated as, given G, modeling the probability distribution of n-token sequences $Y = (y_1, ..., y_n)$ that describe G.

3.1 Framework Overview

Our framework comprises four components: the entity aligner and edge detector, which match Wikipedia sentences with subgraphs; the sentence

Dataset	KG	Text Corpus	Domain	Instances	Entities	Triples	Relations	Star Graphs
WebNLG	DBpedia	Handcraft	15 DBpedia categories	25,298	2,730	3,221	354	57%
AGENDA	-	Scientific abstract	Scientific research	40,720	159,691	177,568	7	**2%**
EventNarrative	Wikidata	Wikipedia	Events	224,428	305,685	649,337	672	94%
TEKGEN	Wikidata	Wikipedia	Open domain	7,895,789	**4,856,439**	11,373,838	663	96%
Ours	Freebase	Wikipedia	Open domain	**8,769,634**	1,853,752	**15,472,249**	**1,724**	22%

Table 1: Comparison of characteristics of the datasets used for graph-to-text generation

trimmer, which refines sentences; and a PLM that is to be fine-tuned, as shown in Figure 1. The coreference resolution module replaces pronouns or aliases with the corresponding entities, for example, "the band" and "this release" would be replaced with "Australian heavy metal band Lord" and "Return of the Tyrant." The wikification module is applied to identify Wikipedia entities in the sentence. These entities are then mapped to Freebase to identify a corresponding subgraph. The triples of the subgraph are used to trim the coreference-resolved sentence. The resulting trimmed sentence and subgraph are fed into a PLM for fine-tuning, creating a graph-to-text model. In the inference stage, the model takes a user-specified subgraph as input and outputs a generated sentence that describes it.

3.2 Graph-Text Alignment

It is impractical to rely on human annotation to write a sufficient number of sentences to describe subgraphs and form a dataset because of the large volume and diversity of entities and relationships in large-scale open knowledge graphs. Our approach is to automatically generate training examples by aligning a Wikipedia sentence, denoted as W, with its corresponding subgraph G to create a graph-sentence pair (G, W).

Edge Detector To find a matched (G, W) pair, it is necessary to first identify the subgraph G from the entire knowledge graph $\mathcal{G}$ based on the entities present in the Wikipedia sentence W. We propose an edge detector that takes a Wikipedia sentence W and $\mathcal{G}$ as input and detects a subgraph G consisting of a set of triples $t_i = (s_i, p_i, o_i)$ where s_i, o_i are entities in the KG and p_i a relation.

The edge detector iterates through all the triples in $\mathcal{G}$ and assigns the appropriate edge containing the KG entity pairs (e_i, e_j) that also appear in the Wikipedia sentence W. If there is only one edge between the entity pair, that edge will be selected. If there are multiple edges between the same pair of entities, the edge detector chooses the edge whose label tokens appear in the sentence as the priority. If more than one edge has label tokens appearing in the sentence, the edge detector selects the edge with the highest frequency in the KG. The detected edges will finally be used to construct the subgraph G. In this way, the collection of subgraphs consists of various shapes, which allows the model to learn the presentation of complex graphs and to be more versatile.

Entity Aligner The edge detector relies on the ability to identify the entity e present in the Wikipedia sentence W. To accomplish this, an entity aligner is needed that matches a knowledge graph entity, $e \in E$, to a Wikipedia token span, $w_{aligned} = [w_k, ..., w_{k+m}]$, within a Wikipedia sentence W. The entity aligner is made up of a WikiTitle-to-KG mapping, wikification (Csomai and Mihalcea, 2008), and coreference resolution (McCarthy and Lehnert, 1995).

To identify the KG entity e from Wikipedia sentence W, we utilized a WikiTitle-to-KG mapping that serves as an intermediary bridge between Wikipedia titles and their corresponding KG entities. This mapping is particularly useful for identifying the token spans that represent KG entities in the sentence. We then utilize hyperlinks that point to Wikipedia titles to achieve wikification. These hyperlinks establish a mapping between the anchor texts in sentences and their corresponding Wikipedia titles. For each Wikipedia article, we collect all hyperlinks and assume that any anchor text matches found within the same article also refer to the same Wikipedia title.

However, the wikification process may still fail to identify some Wikipedia entities, particularly when involving pronouns and aliases. To enhance the coherence and readability of the Wikipedia sentence W, coreference resolution (Zheng et al., 2011) is applied to replace all the Wikipedia token spans with the same entities they refer to. We used AllenNLP's coreference resolution (Gardner et al., 2017; Lee et al., 2017) with default settings.

3.3 Sentence Trimmer

Automatic methods for aligning text corpora with KG inevitably have errors. These errors can be divided into two categories: the sentence W may contain unrelated information that is not present in the graph G, or the sentence may miss some of the information present in G. If a relation in the graph is expressed semantically incorrectly in the sentence, we view it as both a piece of unrelated information and missing information. Using the graph-sentence pairs in the first category as training instances can lead to PLMs generating fabricated facts that are not present in the input graph, a phenomenon known as "hallucination" when fine-tuning for graph-to-text tasks on large-scale, open-domain knowledge graphs. In this work, we focus on mitigating this phenomenon and propose a sentence trimmer that reduces the chance for PLMs to hallucinate facts by removing extra information that is not present in the graph while maintaining the sentence's main idea.

The sentence trimmer will output a trimmed version of the sentence, denoted as W_{trim}, and it involves two steps: identifying the shortest dependency paths (SDPs) among every pair of entities of the graph and trimming the sentence accordingly. We utilized the SDP (Bunescu and Mooney, 2005) to identify the text sequence that describes the relations between entity pairs in the graph. SDP is the shortest path between two tokens in a sentence's semantic dependency parse tree. We used spaCy (Honnibal et al., 2020) for semantic dependency parsing and record the SDPs of every entity pair (e_i, e_j) in the triples t_i in the subgraph G. Figure 2 shows an example of a coreference-resolved sentence's dependency parse tree. According to the subgraph's triples {(`FlyBack`, *software_genre*, `Backup software`), (`FlyBack`, *operating_system*, `Linux`), (`FlyBack`, *basis*, `Git`)}, the three SDPs are (①, ②), (①, ②, ③, ④), and (①, ②, ⑤, ⑥, ⑦), respectively. We analyzed the dependency parse tree of W, defined w_{start} and w_{end} as the leftmost and rightmost tokens in W that also appeared in the SDP of an entity pair. After reviewing all triples, we updated w_{start} and w_{end} to the farthest ones. The resulting token sequence, $W_{trim} = [w_{start}, ..., w_{end}]$, became the trimmed sentence. In the example above, the leftmost token of the three SDPs is "FlyBack", and the rightmost token is "Git", so the trimmed sentence is *"FlyBack is an open-source Backup Software for Linux based on Git."*

3.4 Model

We examined T5 (Raffel et al., 2020) and BART (Lewis et al., 2020), two PLMs, for fine-tuning them on our dataset as PLMs. Both models follow the Transformer encoder-decoder architecture (Vaswani et al., 2017). The main distinctions between the two models are their pretraining method and the text corpora used. The details of the models were explained in (Raffel et al., 2020; Lewis et al., 2020). Our graph is transformed into a token sequence by linearizing it, which is then input into the model. Following the method in Ribeiro et al., 2021, the graph in Figure 1 would be linearized into the token sequence "*<H> Lord (band) <R> artist genre <T> Heavy Metal Music <H> Return of the Tyrant <R> album artist <T> Lord (band) <H> Return of the Tyrant <R> album genre <T> Heavy Metal Music*" where the special tokens <H>, <R> and <T> denote subjects, relations, and objects, respectively.

4 Dataset

4.1 Text Corpus and Knowledge Graph

Text Corpus We used the Wikipedia dump released on Sep. 1st 2019 as our text corpus.[2] The raw dump was in the form of wikitext source and metadata embedded in XML. To preprocess the raw dump, we utilized WikiExtractor, an existing tool.[3] The tool transformed the compressed XML file into multiple plain text files consisting of numerous Wikipedia articles without tables and catalogs.

Knowledge Graph We pre-processed the Freebase dump [4] to be used as a knowledge graph. The pre-processing included removing one of the *reverse triples* in Freebase as they have the same semantic meaning and are redundant in our work and removing *intermediate nodes* from the Freebase dump by concatenating the two edges that connect two entities through the intermediate node.

4.2 Dataset Statistics

We have collected a dataset of 8,769,634 graph-text pairs in total. The dataset includes graphs that contain at most 15 triples and at most 20 entities. The

[2]Released by Wikimedia: https://dumps.wikimedia.org
[3]Released by attardi: https://github.com/attardi/wikiextractor
[4]http://commondatastorage.googleapis.com/freebase-public/rdf/freebase-rdf-latest.gz

Model	BLEU			METEOR			chrF++		
	All	Seen	Unseen	All	Seen	Unseen	All	Seen	Unseen
(Ribeiro et al., 2021)	59.70	64.71	53.67	44.18	45.85	42.26	75.40	78.29	72.25
(Wang et al., 2021)	60.56	66.07	53.87	44.00	46.00	42.00	-	-	-
(Aghajanyan et al., 2021)	56.30	64.80	46.10	42.00	46.00	38.00	-	-	-
(Nan et al., 2021)	61.44	65.82	**56.01**	**45.00**	46.00	**43.00**	-	-	-
TEKGEN (T5-large)	60.43	65.49	54.32	44.06	46.04	41.90	75.73	78.83	70.13
+ trimmer	60.82	65.42	55.12	44.25	46.13	42.18	76.16	79.11	72.35
Ours (T5-large)	60.26	65.44	54.06	44.08	45.90	41.98	75.83	79.02	72.35
+ trimmer	**61.46**	**66.49**	55.35	44.30	**46.23**	42.08	**76.20**	**79.35**	**72.76**

Table 2: Performance comparison of different graph-to-text models on the WebNLG dataset

shape rank	1	2	3	4	5
#sentences	5346453	1635504	281060	222821	198611
shape rank	6	7	8	9	10
#sentences	191230	160739	62109	60874	51872

Table 3: Frequent graph shapes with the count of sentences

logarithmic distribution of sentences that contain a certain number of triples and entities can be seen in Figure 3 and Figure 4. Table 4 shows the number of unique graph shapes that contain a certain number of entities. It is worth noting that the graphs that are isomorphic are considered to have the same shape, meaning that the isomorphism graphs are identical except for the names of the entities and the labels of the edges. We utilized NetworkX to detect isomorphic graphs and discovered a total of 7,920 unique graph shapes in our dataset.

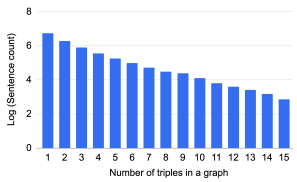

Figure 3: Dataset distribution with number of triples in graphs

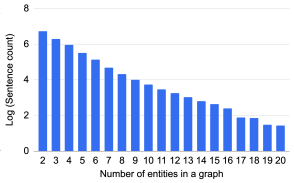

Figure 4: Dataset distribution with number of entities in graphs

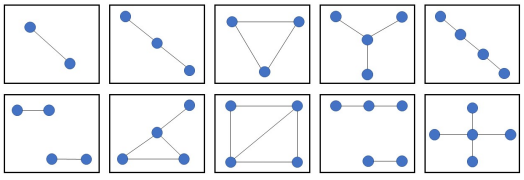

Figure 5: Top 10 most frequent graph shapes in our dataset

Table 1 compares five datasets: WebNLG, AGENDA, EventNarrative, TEKGEN, and ours. The datasets differ in terms of their knowledge graph source, text corpus, domain coverage, number of instances, entities, triples, relations, and percentage of star graphs. Our dataset, which uses Freebase as the knowledge graph and Wikipedia as the text corpus, stands out by covering the open domain with a large number of instances, entities, triples, and relations (8,769,634, 1,853,752, 15,472,249, and 1,724 respectively). It also has a lower percentage of star graphs (22%) and more diverse graph shapes, as shown in Figure 5. It is not limited to star graphs, but also includes triangles, paths, etc.

4.3 Dataset Split

We split the dataset of graph-sentence pairs into 84 distinct domains based on the most frequent domain of the edges in a graph and randomly choose one in cases where all unique domains have the same frequency. We then split these domains into 54 seen and 28 unseen for the purpose of evaluating the model's generalization ability. We further split the graph-sentence pairs of seen domains into 90% training, 5% development, and 5% testing, and all graph-sentence pairs in the unseen domains are put into the test set.

#entities	2	3	4	5	6	7	8	9	10	11
#shapes	1	2	7	23	122	705	1690	1705	1267	830
#entities	12	13	14	15	16	17	18	19	20	sum
#shapes	542	378	222	176	106	58	52	22	12	7920

Table 4: The number of shapes in a graph with entities ranging from 2 to 20.

5 Experiments

5.1 Experiment Setup

We fine-tuned various versions of T5 (small - 60M parameters, base - 220M parameters, and large - 770M parameters) and BART (base - 140M parameters, large - 400M parameters) models on our dataset with a batch size of 8 using Adam Optimizer (Kingma and Ba, 2014) and an initial learning rate of 3*e-5. We utilized a linearly decreasing learning rate schedule without warm-up and set the

maximum target text length to 384 tokens. Our implementation was based on the work of (Ribeiro et al., 2021), which utilized PLMs from Hugging Face(Wolf et al., 2019).

To demonstrate the adaptability of our method across various knowledge graphs, we fine-tuned T5-large and BART-large models on the TEKGEN corpus using identical parameters as on our dataset. However, we were unable to use all instances in the original corpus as the mapping between entity names and their surface texts was unavailable. Instead, we used aliases provided by TEKGEN and regular expressions to match time and people's names. As a result, we obtained 3,811,288 instances for training, 476,439 for development, and 484,958 for testing out of a total of 6,310,061, 788,746, and 796,982 respectively. Furthermore, since our dataset and the TEKGEN corpus are both generated automatically, they do not have a "golden ground truth." To determine if the models fine-tuned on our dataset can enhance performance on handcrafted datasets, we further fine-tuned the T5-large models on the WebNLG 2017 dataset and compared their performance with other existing methods.

Model	BLEU	METEOR	chrF++
BART-large	41.51	23.62	47.13
+ trimmer	48.32	29.90	57.50
T5-large	43.03	24.21	48.05
+ trimmer	**49.83**	**30.52**	**58.25**

Table 5: Performance of BART and T5 on the TEKGEN dataset with or without sentence trimmers

5.2 Evaluation Metrics

We assessed our approach using both automatic and human evaluations. For the automatic evaluation, we present our results using standard natural language generation metrics such as BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), and chrF++ (Popović, 2015).

To measure the quality of the sentences generated by PLMs trained on trimmed sentences compared to those trained on original Wikipedia sentences, we conducted a human evaluation. The human annotators were asked to evaluate the quality of the generated sentences in two aspects: semantics and grammar. We randomly selected 100 graphs from our test set, as well as the generated sentences by the T5-large model fine-tuned on original Wikipedia sentences and trimmed sentences, then shuffled the 200 graph-text pairs. The human annotators were asked to provide scores for the sentences describing the graphs; each of them evaluated all 200 graph-sentence pairs. For semantics, the human annotators were asked to provide the number of *hallucinated entities* (entities not in the graph but expressed in the sentence), *missed entities* (entities in the graph but not expressed in the sentence), *hallucinated relations* (relations not the number of *missed relations* (relations in the graph but not expressed in the sentence). For grammar, human annotators give scores on a scale of 1-5. 5 indicates no errors, 4 indicates one error, 3 indicates two to three errors, 2 indicates four to five errors, and 1 indicates more than five errors.

6 Results and Discussion

Table 6 illustrates the input graphs and the sentences generated by T5-large models trained on both the original Wikipedia sentences and the trimmed Wikipedia sentences. The model trained on original Wikipedia sentences tends to generate sentences that contain information not present in the input graph. This can include fabricated facts like the age of death of Arthur Morry or the renaming of the US Naval Academy during World War II. The third graph example also shows that the model fabricated that Goldie Gets Along is a comedy and was filmed in 1951, but in reality, the movie was filmed in 1933. Our approach decreases the chances of PLMs fabricating facts by focusing only on the information present in the input graph.

Table 7 and 5 show the automatic evaluation results of BART and T5 models fine-tuned with and without the sentence trimmer on our dataset and the TEKGEN dataset. The BLEU, METEOR, and chrF++ metrics consistently improve with the sentence trimmer by more than 12% on our dataset and 7% on the TEKGEN dataset, indicating that the PLMs fit better on the trimmed sentences. The trimmed sentences contain less information that is not present in the input graphs, reducing the probability of the PLMs fabricating entities/relations and improving the automatic evaluation metrics. It is worth noting that the PLMs perform better on the TEKGEN dataset than on our dataset. One possible explanation is that our dataset is more complex for PLMs to learn due to the diverse shapes of the graphs.

Table 2 compares the performance of the different graph-to-text models on the WebNLG dataset. We re-print the results of models in the other four

Input Graph	w/o Sentence Trimmer	w/ Sentence Trimmer
(Arthur Morry, place of death, Brisbane)	Arthur Morry died in Brisbane, Queensland, aged 79.	Arthur Morry died in Brisbane.
(Annapolis Maryland, location contains, US Naval Academy)	During World War II, the US Naval Academy in Annapolis, Maryland was renamed the US Naval Academy in Annapolis, Maryland, and the US Naval Academy in Annapolis, Maryland was renamed the US Naval Academy in Annapolis, Maryland.	US Naval Academy in Annapolis, Maryland.
(Goldie Gets Along, film directed by, Malcolm St. Clair filmmaker) (Goldie Gets Along, film performance actor, Lili Damita) (Goldie Gets Along, film performance actor, Charles Morton actor)	Goldie Gets Along is a 1951 American comedy film directed by Malcolm St. Clair (filmmaker) and starring Lili Damita and Charles Morton (actor).	Goldie Gets Along was directed by Malcolm St. Clair (filmmaker) and starred Lili Damita and Charles Morton (actor).

Table 6: Comparison of generated sentences with and without sentence trimmer for selected input graphs

Model	BLEU			METEOR			chrF++		
	All	Seen	Unseen	All	Seen	Unseen	All	Seen	Unseen
BART-base	33.18	33.33	27.52	17.18	17.26	14.63	36.56	36.74	30.75
+ trimmer	**46.49**	**46.77**	36.67	24.43	24.53	21.30	49.92	50.12	43.29
BART-large	32.35	32.48	27.56	17.45	17.53	15.07	37.12	37.29	31.58
+ trimmer	46.04	46.18	40.98	24.35	24.41	22.17	49.69	49.85	44.72
T5-small	19.48	19.53	17.34	15.78	15.85	13.79	33.92	34.08	28.90
+ trimmer	43.72	43.87	38.11	23.40	23.48	21.10	48.15	48.31	42.65
T5-base	16.89	16.95	14.63	16.23	16.30	14.10	35.37	35.54	29.84
+ trimmer	42.18	42.29	37.85	24.20	24.27	21.94	49.63	49.80	44.18
T5-large	22.22	22.26	20.41	17.16	17.23	15.02	36.78	36.95	31.40
+ trimmer	45.12	45.16	**43.40**	**24.77**	**24.84**	**22.54**	**50.44**	**50.60**	**45.21**

Table 7: Automatic evaluation results on our dataset

Model	#Entities hallucinated	#Entities missed	#Relations hallucinated	#Relations missed	Grammar
T5-large	1.643	0.063	1.363	0.240	4.613
+ trimmer	0.260	0.056	0.300	0.370	4.356

Table 8: Human evaluation result of T5-large model performance on 200 graph-text pairs

papers. As shown in the table, models with the sentence trimmer consistently perform better than those without the trimmer on both the TEKGEN and our dataset, particularly in terms of BLEU and chrF++ metrics. The T5 model fine-tuned on our dataset with the sentence trimmer outperforms most other models based on fine-tuning PLMs in all metrics, except for the model reported in (Nan et al., 2021). This may be due to the fact that (Nan et al., 2021) utilizes human-written sentences, which are of higher quality. This highlights the importance of high-quality data, specifically graph-sentence pairs, for fine-tuning PLMs in the graph-to-text task. Therefore, better graph-sentence alignment methods are needed to improve the performance of graph-to-text tasks.

Table 8 shows averages of the human annotators' evaluation scores of generated sentences from 100 randomly selected graphs by T5-large models fine-tuned on original Wikipedia sentences and trimmed sentences. It can be observed that the T5-large model generated sentences that miss very few entities or relations in the graph, as the average number of missed entities and relations is less than 0.07 and 0.38 respectively, which are much lower than 1. The model trained on the trimmed sentence has 1.4 and 1.0 fewer entities and relations respectively, which indicates that our sentence trimmer reduces the chance of hallucinations in PLMs when conducting graph-to-text tasks. The grammar scores of sentences generated by the model trained on the trimmed sentences are slightly worse than that of the model trained on the original Wikipedia sentences, but they are still acceptable as the average number of grammar errors is less than one, with a score of 4 indicating that the sentence has less than one grammar error.

7 Conclusion

In this paper, we propose an approach to address the problem of graph-to-text generation from large-scale, open-domain knowledge graphs. We aim to increase the diversity of training instances collected automatically and to alleviate the problem of hallucination when fine-tuning PLMs for graph-to-text generation tasks. This improves the credibility of generated sentences by PLMs on the task.

Limitations

This paper presents a method and a new dataset for generating natural language descriptions of knowledge graphs. However, it requires a mapping between the knowledge graph entities and Wikipedia entities. If no such mapping exists, the method cannot be applied. Additionally, the Sentence Trimmer approach proposed in this paper can alleviate the problem of "hallucination" in PLMs, but it may result in grammatical errors. Furthermore, the method focuses on describing the content of the input graph without considering context entities/relations in the KG, which may make the generated sentences less natural. The method also doesn't handle mediator nodes in knowledge graphs, but this is an area of future research.

Ethics Statement

All the data sources used in the data collection process are publicly available. All human annotators have been paid.

References

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

Armen Aghajanyan, Dmytro Okhonko, Mike Lewis, Mandar Joshi, Hu Xu, Gargi Ghosh, and Luke Zettlemoyer. 2021. HTLM: Hyper-text pre-training and prompting of language models. *arXiv preprint arXiv:2107.06955*.

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.

Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731.

Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg, and Emiel Krahmer. 2019. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 552–562.

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

Anthony Colas, Ali Sadeghian, Yue Wang, and Daisy Zhe Wang. 2021. Eventnarrative: A large-scale event-centric dataset for knowledge graph-to-text generation. In *Proceedings of the 35th Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Andras Csomai and Rada Mihalcea. 2008. Linking documents to encyclopedic knowledge. *IEEE Intelligent Systems*, 23(5):34–41.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017a. Creating training corpora for NLG micro-planning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 179–188.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017b. The WebNLG

challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform. In *arXiv preprint arXiv:1803.07640*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python.

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

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.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. 2009. Dependency parsing. *Synthesis lectures on human language technologies*, 1(1):1–127.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197.

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.

Diego Marcheggiani and Laura Perez-Beltrachini. 2018. Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9.

Joseph F McCarthy and Wendy G Lehnert. 1995. Using decision trees for coreference resolution. *arXiv preprint cmp-lg/9505043*.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, et al. 2021. Dart: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447.

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.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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 FR 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.

Leonardo FR Ribeiro, Yue Zhang, Claire Gardent, and Iryna Gurevych. 2020. Modeling global and local node contexts for text generation from knowledge graphs. *Transactions of the Association for Computational Linguistics*, 8:589–604.

Martin Schmitt, Leonardo FR Ribeiro, Philipp Dufter, Iryna Gurevych, and Hinrich Schütze. 2021. Modeling graph structure via relative position for text generation from knowledge graphs. In *Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing*, pages 10–21.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27.

Bayu Distiawan Trisedya, Jianzhong Qi, Rui Zhang, and Wei Wang. 2018. GTR-LSTM: A triple encoder for sentence generation from RDF data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1627–1637.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Qingyun Wang, Semih Yavuz, Xi Victoria Lin, Heng Ji, and Nazneen Rajani. 2021. Stage-wise fine-tuning for graph-to-text generation. In *Proceedings of the ACL-IJCNLP 2021 Student Research Workshop*, pages 16–22.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Jiaping Zheng, Wendy W Chapman, Rebecca S Crowley, and Guergana K Savova. 2011. Coreference resolution: A review of general methodologies and applications in the clinical domain. *Journal of Biomedical Informatics*, 44(6):1113–1122.