# UTILIZING LANGUAGE MODELS FOR SYNTHETIC KNOWLEDGE GRAPH GENERATION

**Shuran Fu**
NUS Research Institute
shuran.fu@nusricq.cn

**Peihua Mai**
National University of Singapore
e0545964@u.nus.edu

**Yan Pang***
National University of Singapore
jamespang@nus.edu.sg

## ABSTRACT

Knowledge Graphs play a pivotal role in various machine-learning tasks. However, constructing these datasets is challenging due to their semantic and structural complexity, often resulting in limited data size. Synthetic graph generation has been applied to augment graph datasets and has proven beneficial in domains such as social network analysis and recommendation systems. Despite this, generating graphs with extensive textual attributes remains underexplored. Large language models (LLMs) possess the capability to generate text and reason about complex data structures, including graphs. In this paper, we leverage the generative and reasoning abilities of LLMs to propose a novel framework for synthetic knowledge graph generation. Our framework integrates two transformers and a text data augmentation module, where prompt and fine-tuning approaches are used to generate sentences and Mahalanobis distance is applied to measure outliers. This framework offers straightforward application and high flexibility, which can effectively generate graph datasets that have a similar triple distribution with the real one. We combine the generated data with real data by either concatenation or mixture way and through extensive experiments on downstream tasks, we demonstrate the effectiveness and versatility of our approach.

## 1 INTRODUCTION

Generative models, such as Generative Adversarial Networks (GAN) Goodfellow et al. (2014) and Variational Autoencoders (VAE) Kingma & Welling (2013), have gained more interest in the last few years due to their capacity to generate synthetic data with similar distributions as real data. In some industries, real data can be difficult to achieve because of privacy concerns or technical barriers. For instance, patients and medical data are usually considered to be private, thus hindering the training and applications of artificial intelligence tools in the healthcare industry. Synthetic data could be a potential solution for this data scarcity problem and act as a great supplement for real datasets. It could be applied to train the models for down-streaming tasks without sacrificing the models' efficiency and accuracy that only use real data.

Graph is a type of data structure that can efficiently store information and describe the relations between different entities. Compared with text and numeric datasets, synthetic graph datasets are more difficult to generate and evaluate due to their semantic and structural complexity. Before the emergence of deep learning methodology, some algorithms were proposed to generate synthetic graphs. In the Erdös-Rényi Model ERDdS & R&wi (1959), a graph is chosen uniformly and randomly from a set of all possible graphs with a predefined number of nodes and edges. The variant of this model was introduced in Janson et al. (2011), where a graph is generated by connecting nodes randomly with a fixed probability $p$. However, it is found that it violates the power laws commonly existing in social networks. Barabási–Albert Method is a typical method proposed in Albert

*Corresponding Author

& Barabási (2002), which follows a preferential attachment idea where new nodes prefer to connect with existing nodes. Another type of model is structure-driven, where structural indicators are used to generate new graphs Leskovec et al. (2010); Mahadevan et al. (2006). However, these algorithms are all criticized for some drawbacks, such as the inability to imitate real-world phenomena and high computation costs, especially when the nodes and edges in the graphs have their attributes.

Deep learning-based generators were proposed in some papers You et al. (2018); Liao et al. (2019) and they applied autoregressive models to generate graphs. Graph Neural Network was also used in Grover et al. (2019), where the model parameterizes variational autoencoders (VAE) with graph neural networks and uses a novel iterative graph refinement strategy inspired by low-rank approximations for decoding. In Agarwal et al. (2023), the graph generator could generate a variety of benchmark datasets accompanied by ground-truth explanations, which can be helpful for the evaluation of Graph Neural Network explainability. However, most methodologies lack the ability to generate features in addition to structures, thus for some types of graphs where features conclude extensive text information, it is still difficult to generate datasets that can be directly applied to downstream tasks.

The development of large language models (LLM) is the most noteworthy topic in the field of natural language processing in recent years, which brings a blowout for machine-generated text research. Large language models are capable of generating synthetic text and also solving different types of tasks Radford et al. (2019), such as mathematical problems solving Imani et al. (2023) and code writing Chen et al. (2021). The interaction and reasoning capabilities of the language model empower itself with the potential for solving more complicated tasks, including understanding the graph structure data Li et al. (2023).

For some graph problems, particularly those containing nodes with text attributes, the LLM can be introduced in the task-solving procedure to improve graph learning. There are multiple ways that Graph Neural Networks (GNN) can be integrated with LLM to capture the structural and contextual information, which can mainly be divided into LLM as enhancers, LLM as predictors and GNN-LLM Alignment. In the first type of design, the language models are mainly used to enhance the quality of node embeddings that would be fed into the GNN He et al. (2023); Chen et al. (2024); Chien et al. (2021). When the language models are used as predictors, a graph is flattened into a sequence and then is fed into the language model as input to generate predictionsWang et al. (2024); Zhao et al. (2023); Chai et al. (2023). GNN-LLM alignment designs ensure that each encoder's unique functionalities are preserved while coordinating their embedding spaces at a specific stage Radford et al. (2021); Yang et al. (2021).

Though previous research has proved that language models are capable of understanding graph data structures and attributes, it is still not revealed whether they can act as an important part of the synthetic graph generation procedure. Motivated by this question, we explored how language models can be utilized to assist in synthetic graph dataset generation, especially those containing relationships with semantic information. The contribution of this paper can be divided into the following aspects: (a) We proposed **a novel framework to generate synthetic knowledge graph datasets**, which can be used in downstream graph tasks. This paradigm has a straightforward design, which is simple to follow. (b) We provided **a template pipeline of text augmentation, transformation and cleaning**. We use Mahalanobis distance to effectively remove textual outliers and sample synthetic datasets with similar triple distribution to real datasets. (c) Through extensive experiments, we show that the synthetic dataset can be introduced into down-streaming tasks, which provided a promising application for this research.

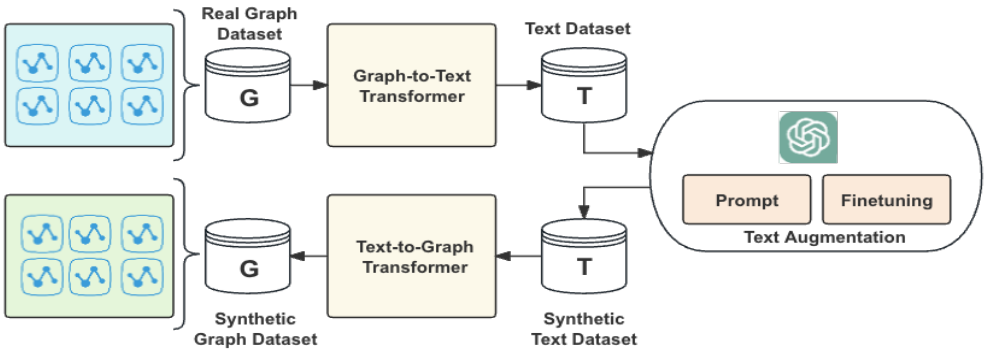


Figure 1: Overview for the framework of the synthetic Knowledge graph generation

## 2 Related Work

### 2.1 KG-to-Text Generation

Recent works have utilized graph neural networks (GNN) Velickovic et al. (2017) and pretrained language models (PLM) Lewis et al. (2019); Raffel et al. (2020) to generate fluent text consistent with KG inputs. Studies on KG-to-text generation falls into three directions: 1) Graph transformer: traditional methods train a transformer-based model with a modified encoder module that effectively captures a graph's structure information Koncel-Kedziorski et al. (2019); Schmitt et al. (2021); Guo et al. (2019); Ribeiro et al. (2020). 2) Pre-trained models: researchers have adapted and finetuned PLMs for KG-to-text task and obtained better results than graph transformer-based approaches in some cases Ribeiro et al. (2021); Chen et al. (2020); Kale & Rastogi (2020). 3) Integration of PLM and Graph-aware modules: recent works further improve the performance by fusing graph-aware modules into PLM Colas et al. (2022); Ke et al. (2021).

### 2.2 Text-to-KG Generation

Earlier works addressed text-to-kg generation as multi-stage models, consisting of subtasks such as entity extraction Martins et al. (2019) and relationship classification Zeng et al. (2014); Zhang et al. (2017). Grapher Melnyk et al. (2022) first generates graph nodes using PLM, followed by an edge construction head for efficient KG construction. These models require complex task-specific designs and present poor flexibility for different domains. To overcome the limitation, another direction of works frames the KG generation as a sequence-to-sequence (seq2seq) problem, where the PLMs are finetuned to generate the linearized graph in an end-to-end manner Lu et al. (2022); Cabot & Navigli (2021); Dognin et al. (2021). ReGen Dognin et al. (2021) leverages reinforcement learning to finetune the PLM with a seq2seq generation objective, achieving state-of-the-art performance on WebNLG+ 2020 datasets. Powered by the advances of large language models (LLMs), recent works prompt LLMs to generate relational tripplets in a zero/few-shot setting Wei et al. (2023); Wadhwa et al. (2023).

### 2.3 Text Augmentation

Text augmentation is the technique of increasing training data diversity without directly collecting more data, and most text augmentation methods deal with text input with data space, which transforms text data by its raw form. At the character level, Belinkov and Bisk Belinkov & Bisk (2017) add different kinds of noise to the training data, and Coulombe Coulombe (2018) implements rule-based transformations by regular expressions. At the word level, Wei and Zou Wei & Zou (2019) propose Easy Data Augmentation, a set of token-level random perturbation operations including insertion, deletion, and swap. Some researchers also proposed methods to replace words by word embeddings Alzantot et al. (2018); Li et al. (2017), and language models can be utilized to perform the substitution Hu et al. (2019); Jiao et al. (2019). In Shi et al. (2021), they provided

an augmentation technique at the phrase level, where they substitute substructures of the sentence. Back translation is one of the most popular methodologies when the whole sentence is taken into consideration Sennrich et al. (2015); Xie et al. (2020). Generative models are also widely used in the generation such as GPT-2 Liu et al. (2020); Anaby-Tavor et al. (2020).

# 3 PROPOSED METHOD

## 3.1 PROBLEM FORMULATION

We first introduce the formulated problem in this section, and then we propose our framework to solve the problem in Section 3.2. The generated synthetic datasets will be evaluated in different down-streaming tasks and the details for the experiments are shown in Section 4. In the end, limitations and conclusions are discussed in Section 5.

Our goal is to generate some synthetic graph datasets that can be used as a supplemental or augmented dataset for down-streaming tasks. In some cases, it can also be mixed with real data. Given the previous research, language models can be used in different ways when solving graph problems with different types and features. To fully explore the language models' capacity to generate graphs where nodes and edges have rich textual information and such information is important in the following tasks, we select knowledge graph as our experimental type.

Given a knowledge graph dataset $\mathcal{G}$ in which each data sample can be formally expressed as $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0, \mathcal{R}_0)$, where nodes are denoted as $v \in \mathcal{V}_0$ and labeled edges are represented as $(v_i, r, v_j) \in \mathcal{E}_0$, with $r \in R_0$ denoting as the relation type, we expect the output to be a synthetic dataset, where the nodes in the graph have similar complexity of text attributes and distribution of structure. It can be used in down-streaming tasks and achieve about the same level of performance in terms of the evaluation metrics.

## 3.2 GENERATION FRAMEWORK

There are three main components in the framework: Graph-to-Text (G2T) Transformer, Augmentor and Text-to-Graph (T2G) Transformer. Those transformers will be fine-tuned at first, and when a synthetic dataset is required, the real dataset $\mathcal{G}$ will be transformed into a text dataset by the G2T module, where each sample (a sentence or a paragraph) corresponds to one graph in the real dataset. The Augmentor can generate synthetic text by either fine-tuning or prompting, and the generated text dataset will be cleaned, selected and evaluated by its fidelity and semantic coherence. It will also be compared with the input text dataset. Finally, the text will be transformed into a graph in the T2G module. The overview of the framework is shown in Figure 1.

### 3.2.1 G2T TRANSFORMER

In the G2T module, we followed the design of a state-of-art methodology named **GAP** Colas et al. (2022). The advantage of this model is that it introduced a graph-aware attention module to effectively capture the relationships between nodes and edges. A graph is first linearized into a sequence with all the triples in the graph, divided and concatenated by tags that separate each component in the triple. The tags include head, relation and tail. For example, a triple list {["Harry Potter", "author", "JK Rowling"], ["JK Rowling", "country", "UK"]} will be linearized into "<head>Harry Potter <relation>author <tail>JK Rowling <relation>country <tail>UK".

A transformer encoder will be used to contextualize the vector representations. The self-attention module will be set as the first part which acts as a Global Attention to capture the semantic relationships between all tokens. The $l$-th layer of the module can be formulated as

$$X_l = Attn(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{1}$$

where $W$ are the model parameters with a size of $d_k \times d_k$, and the query, key, and value are computed via $Q = X_{l-1}W_l^Q$, $K = X_{l-1}W_l^K$, and $V = X_{l-1}W_{l-1}^V$. $X_{l-1} \in \mathbb{R}^{n \times d}$ denotes the collection of vectors of the graph's tokens. The $d_k$ is the dimension of word vectors.

Then a graph-aware attention module will be introduced as the second part, which can be formulated as

$$\tilde{X_l^g} = Attn_{M,T}(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}} + M + \gamma(T)\right)V. \tag{2}$$

The $Q, K, V$ in Equation (2) are constructed from $X_g$, where $X_g = pooling(X_l), X_g \in \mathbb{R}^{m\times d}$. $m$ is the number of graph components. This pooling layer can return a vector that is unique for each entity for the node or relation for the edge, even if it appears several times in the linearized sequence. $M \in \mathbb{R}^{m\times m}$ is a masking matrix and $\gamma(T)$ is a type encoding matrix where $T \in \mathbb{R}^{m\times m}$ encodes the information of connection type in the matrix. And finally, the $l$-th layer will generate the output as

$$\tilde{X}_l = gather(\tilde{X_l^g}) + X_l, \tag{3}$$

where the word representations from the graph representations generated by Equation(2) are gathered and are added as residual to $X_l$. Ultimately, the encoded representations will be decoded into a sequence by a decoder.

#### 3.2.2 TEXT AUGMENTATION MODULE

For the augmentation module, we implemented two different ways to generate synthetic text: prompt engineering and fine-tuning. Prompts can effectively elicit knowledge from language models Shin et al. (2020); Jiang et al. (2020) and the design of in-context learning can also take advantage of the few-shot learning and inference ability of the models Xie et al. (2021), thus we provide another type of augmentation method by prompting the LLM to generate text. Given the text dataset $D_r = \{d_i\}_{i=1}^{D_N}$, we draw a sample $D_s$ as in-context learning examples. Denote $p$ as the designed prompt, we can formulate the generation procedure as

$$\text{Prompting}: \qquad d_{new} = f_{LLM}(D_s, p). \tag{4}$$

In the fine-tuning approach, for the graph dataset we owned, the validation texts corresponding to graphs are first extracted, followed by the construction of a new dataset $D_g$, where $D_g = \{d_i\}_{i=1}^{D_N}$. Each $d_i$ in $D_g$ is a sentence or a paragraph. Then we generate a list of sequences $S_g = \{S_j\}_{j=1}^{S_N}$ in order, and each sequence consists of a fixed number of sentences in the dataset. For example, the first sequence can be denoted as $s_1 = [d_1, d_2, ...d_\alpha]$, where $\alpha$ is the number of sentences in each sequence and $S_N = \frac{D_N}{\alpha}$. Then an instruction $I$ for generating samples of synthetic text is paired with the concatenation of the sequence to form the fine-tuning sample. After the model is fine-tuned, and given the same instruction as before, the model is capable of generating a group of new sentences $d_{new}$. The typical pipeline is as follows:

$$\begin{aligned} &\text{Finetuning}: \qquad \mathbf{F} = f_{LLM}(S_g, I) \\ &\text{Generation}: \qquad d_{new} = Parse(\mathbf{F}, I), \end{aligned} \tag{5}$$

where $\mathbf{F}$ is the fine-tuned language model.

#### 3.2.3 T2G TRANSFORMER

For the Text-to-Graph Transformer, similar to the state-of-art architecture **Grapher** proposed in Melnyk et al. (2022), we selected the *text-nodes-class-edges* as the design of the base generator since it has the best performance in terms of evaluation metrics including F-1 score compared with other designs revealed by the paper. The advantage of this design is that the sparse distribution of connection between different nodes is considered, which can more properly reflect the real situation.

Different from the G2T transformer, which considers nodes and edges at the same time, in this transformer, input training text sequences are at first translated into a sequence of nodes, separated with special tokens, "<PAD>$node_1$<NODE_SEP>$node_2$ ... </S>", where $node_i$ represents one

or more words. The <NODE_SEP>is set as the boundary for different nodes, and then the string corresponding to each node is greedy-decoded to generate features for the next step, the edge classifier. The classifier will predict if there is an edge connection between two different nodes based on their features. Since in general, the number of actual edges is small, and the distribution for the edges is imbalanced, predicting edges for all pairs of nodes can be really time-consuming and unnecessary. Thus the Focal loss is set as the objective function,

$$\mathcal{L}_f = -(1 - p_t)^{\lambda} log(p_t), \tag{6}$$

where $p$ is a probability corresponding to a single edge and $t$ is the target class, $\gamma \geq 0$ is a weighting factor, such that $\gamma = 0$ makes Equation (6) equal to traditional cross-entropy loss. The shift of function can improve the accuracy of the edge classifier, and more details for the design are discussed in Melnyk et al. (2022).

By setting the overall framework, our methodology is capable of generating more graphs with unseen entities and relationships, which can be learned from downstream models. Thus this feature differs us from CycleGTGuo et al. (2020), where they also introduced graph-to-text and text-to-graph tasks simultaneously. Given no parallel pairs of text and graph, their work aims to jointly learn the models in a cycle framework by iterative training, however, the goal of our work is to augment the graph datasets.

## 4 EXPERIMENTS

In this section, we first introduce the fundamental settings of the model, and then we provide a preliminary analysis of the generated text dataset, followed by a comparison of the results of the generated graph performance on downstream tasks and an exploration of how to use the synthetic data. We emphasize here that the goal of this paper is to generate synthetic knowledge graphs that can be used as supplementary training data for downstream tasks that achieve comparable performance with initial real data, and there are none of synthetic knowledge graph baselines, so we mainly compared with existing tasks with real data.

### 4.1 FUNDAMENTAL SETTINGS

In the generation stage, we use the predictions on the test set of the data based on the graph-to-text module with a sample size of 1,600. For the prompt approach, we set the size of the in-context learning sample $D_s = 20$ and design different prompts, such as "Given some sentences that describe some facts as follows, generate the same number of other sentences with similar language complexity and sentence length range to describe facts". For the fine-tuning approach, we split three versions and respectively set $\alpha = [1, 5, 10]$ as the number of sentences in each sample of sequence. We utilized `gpt-3.5-turbo` provided by OpenAI for the augmentation and Figure 2 is an illustration of the approaches.

### 4.2 DATA GENERATION AND SELECTION

In each setting mentioned above, we generate around 8,000 sentences, which is fivefold of number of input sentences. To decrease the negative impact of outliers and select the sentence that has a smaller distance to the input dataset distribution, inspired by the outlier detection in Ghorbani (2019), we applied Mahalanobis distance as the metric to evaluate the generated texts.

The Mahalanobis distance can be used as a measure of the distance between a data point $P$ and a distribution $D$. It is a multi-dimensional generalization of the idea of measuring how many standard deviations away $P$ is from the mean of $D$, and it corresponds to standard Euclidean distance in the transformed space if the principle axes are re-scaled to unit variance. Textual outliers can also be detected by this measure and its variants Podolskiy et al. (2021).

The Mahalanobis distance of an observation $\mathbf{x} = (x_1, x_2, x_3..., x_n)^T$ from a set of observations with mean $\mu = (\mu_1, \mu_2, \mu_3..., \mu_n)^T$ and covariance matrix $\mathbf{S}$ is defined as:

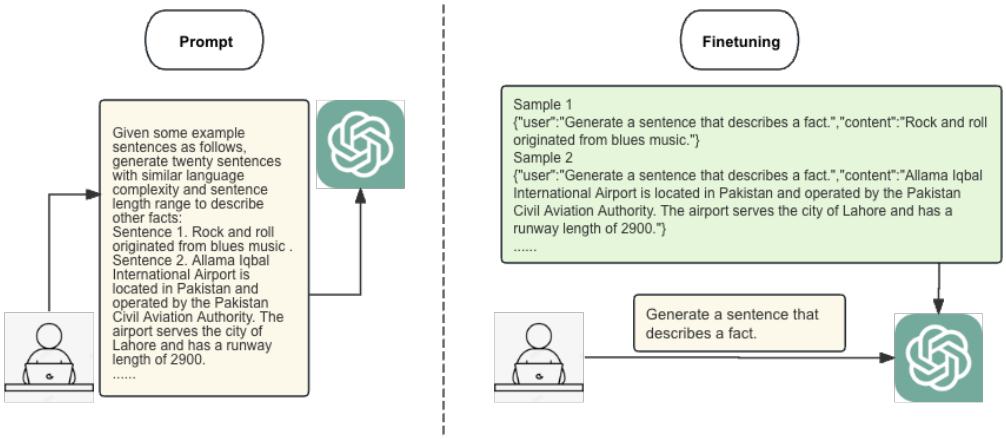


Figure 2: Overview of the Prompt and Fine-tuning approach for text augmentation

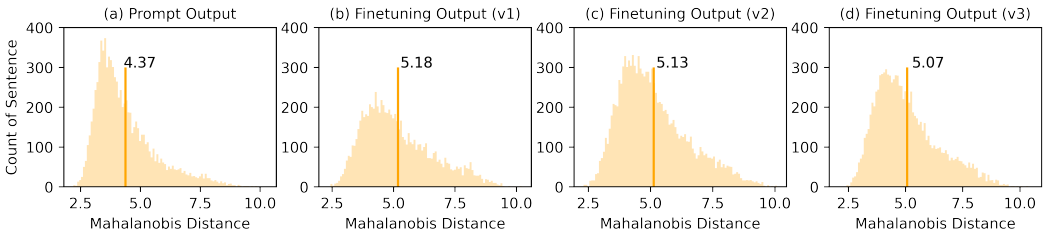


Figure 3: Histogram of Generated Sentence Embedding Mahalanobis Distance

$$D_M(\mathbf{x}) = \sqrt{(\mathbf{x}-\mu)^T \mathbf{S}^{-1} (\mathbf{x}-\mu)} \tag{7}$$

In our work, we constructed the distribution matrix of the input dataset with a shape of $n_l \times d_l$, where $n_l$ is the number of sentences in the input dataset and $d_l$ is the dimension of sentence embedding of the last layer. Since we calculated the embedding by the BERT model provided by huggingface, here $d_l$ equals 768. Then we compute the mean and covariance matrix of the distribution matrix and evaluate the distance between the embedding of the generated sentence and the distribution matrix by (7).

Figure 3 shows the histogram of the distance for the four different settings with the number of bins equal to 100. Figure 3(a) is the distribution of distance from the generated sentence embedding to the input dataset embedding matrix under the prompt approach, and (b), (c) and (d) are the distributions of distance under the finetuning approach with the hyperparameter $\alpha = [1, 5, 10]$ respectively to control the sequence length of the fine-tuning sample. We split the settings in this way to see if the granularity of input will make an impact on the generated text.

The mean distances are labeled by the vertical orange lines in the graph. For the finetuning method, as the setting of $\alpha$ gets larger, the mean distance becomes smaller, though they are all higher than the mean distance under the prompt method with the smallest value of 4.37. The distributions of distance show the same left-skewed feature with different levels, suggesting that the semantic space is not uniformly dense and that removing outliers is a necessity. Thus we decide to remove 10% of the generated texts with the largest distance, which are the data points distributed on the right side in each subfigure.

In the next step, the selected texts are fed into the text-to-graph module to generate graphs. We remove all the graphs containing edges labeled as failed to infer and for each setting, we sample a final dataset with a size of 5,000 that can be experimented with in downstream tasks.

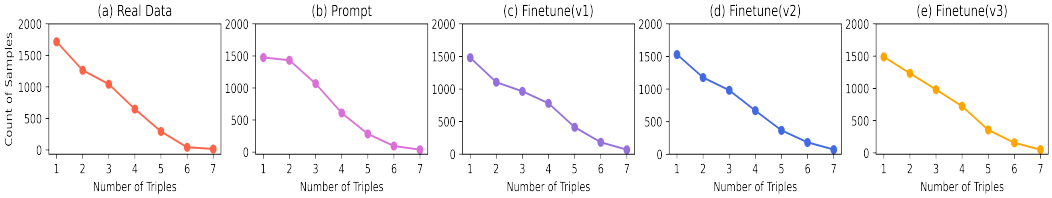


Figure 4: Distributions for Number of Triples in Real and Synthetic Dataset

Table 1: F-1 Score (%) on Real Test Set for Named Entity Recognition Task Under PFN

| Case | #Real Data | #Synthetic Data | Prompt | Finetune-v1 | Finetune-v2 | Finetune-v3 |
|---|---|---|---|---|---|---|
| | 5000 | 0 | | | 97.39 | |
| 1 | 5000 | 5000 | 97.35 | 96.51 | 97.20 | 97.30 |
| 2 | 4000 | 1000 | 96.64 | 96.61 | 96.60 | 96.44 |
| 3 | 3000 | 2000 | 93.75 | 94.15 | 95.01 | 94.83 |
| 4 | 2000 | 3000 | 84.83 | 90.56 | 91.40 | 91.24 |
| 5 | 1000 | 4000 | 81.59 | 83.11 | 85.34 | 85.44 |

Figure 4 shows the distributions for the number of triples in the real and synthetic training set. The x-axis is the number of triples that can be extracted from the given sentence, which ranges from one to seven, and the y-axis is the count of samples. The four synthetic datasets display a similar distribution to the real dataset that the count of sentences increases as the number of triples increases, so the structure complexity of synthetic graphs is similar to the real one.

### 4.3 Named Entity Recognition

Given a piece of text, named entity recognition (NER) aims to extract all the entities in the text. It is a sub-task in graph formulation and provides a supplement for other tasks such as text summarization and question answering. We select Partition Filter Network (PFN)Yan et al. (2021) as the evaluation method as it provides a state-of-the-art result for this task.

Table 1 shows the results of the F-1 score (%) on the real test data for this task. When we only add the synthetic graphs as the supplement, the final score would not change radically, which means the downstream models are capable of learning the samples newly introduced without being negatively influenced by the noise (Case 1).

Next, we consider different ratios of real data and synthetic data when they are combined to train the models. Controlling the size of the training set as the same, when the number of synthetic data introduced becomes larger, the final F-1 score would gradually decrease. For different generation approaches, the mixture ratios are different when the scores decline to some thresholds. When the texts are generated by the fine-tuning method, related graphs could still have a score of over 90% when the size of synthetic data is slightly larger than real data (Case 4).

The mixture proportions would affect the performance of downstream tasks because though we clean and select the augmented text to decrease the sentence embedding distance from the real data, it still contains semantic differences. The text-to-grapher module would infer new types of entities and relationships, for example, text generated by the fine-tuning (v1) method is transformed into a graph dataset with 361 types of relationships, which is larger than the original training data. Furthermore, the synthetic graphs are only mixed into the training set, without integrating into the validation and test set to improve model performance.

In addition, the introduction of synthetic data would help to increase the rate at which the value of the loss function decreases, thus it will converge faster than the case when only real data is used. Figure 5 shows the changes of loss in the first 10 epochs for Case 1. Subfigure (a) compares training and validation loss of real data versus real data augmented by graphs generated from prompt text, and subfigure (b) compares training and validation loss of real data versus real data augmented by

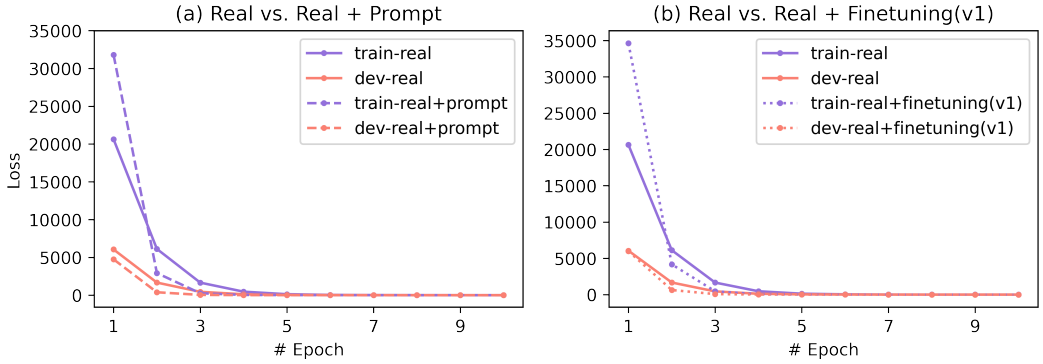


Figure 5: Training and Validation Loss with # Epochs

graphs generated by text from the fine-tuning model. For both cases, the training and validation loss based on the augmented dataset would converge to a lower value in the earlier epochs.

### 4.4 Joint Entity and Relation Extraction

In this section, we explore the impact of the introduction of the generated synthetic graphs on joint entity and relation extraction. It differs from named entity recognition in that it aims to extract not only the entities but all the triples from the given sentences. We use the Set Prediction Networks (SPN) proposed by Sui, et al Sui et al. (2023) and combine the generated dataset with the real WebNLG data to train the models. Table 2 provides the results of the F-1 score (%) on the real test data for the entity recognition, and Table 3 provides the results of the F-1 score (%) for the relation extraction.

Overall, the graphs generated by the finetuned models achieved higher scores on both sub-tasks compared with those generated by prompts. If the synthetic dataset is directly combined to train without removing any real data sample (Case 1), the indicators show a similar level. The score of entity recognition for the data generated by the Finetune-v3 model is even higher than the baseline by 0.64 percent. If the synthetic dataset is mixed with the real dataset for different proportions, for the prompt method, there will be a prominent drop in the score when the ratio between real data and synthetic data changes from 3:2 to 2:3. However, for the finetuning method, the largest decline of score usually happens when the ratio changes from 2:3 to 1:4. It could be considered as the signal that the graphs inferred from text generated by fine-tuned models have the better consistency with real data than those from prompts.

Table 2: F-1 Score (%) on Real Test Set for Entity Recognition Under SPN

| Case | #Real Data | #Synthetic Data | Prompt | Finetune-v1 | Finetune-v2 | Finetune-v3 |
|---|---|---|---|---|---|---|
| | 5000 | 0 | | | 96.60 | |
| 1 | 5000 | 5000 | 96.56 | 96.26 | 96.25 | 97.24 |
| 2 | 4000 | 1000 | 95.85 | 96.16 | 96.91 | 97.01 |
| 3 | 3000 | 2000 | 92.61 | 95.60 | 97.95 | 96.33 |
| 4 | 2000 | 3000 | 88.66 | 93.93 | 96.20 | 95.30 |
| 5 | 1000 | 4000 | 86.78 | 93.87 | 92.34 | 92.17 |

Table 3: F-1 Score (%) on Real Test Set for Relation Extraction Under SPN

| Case | #Real Data | #Synthetic Data | Prompt | Finetune-v1 | Finetune-v2 | Finetune-v3 |
|---|---|---|---|---|---|---|
| | 5000 | 0 | | | 87.06 | |
| 1 | 5000 | 5000 | 86.70 | 86.06 | 86.09 | 85.96 |
| 2 | 4000 | 1000 | 86.64 | 87.00 | 85.88 | 85.92 |
| 3 | 3000 | 2000 | 82.07 | 83.28 | 82.88 | 82.35 |
| 4 | 2000 | 3000 | 69.86 | 78.07 | 79.56 | 79.01 |
| 5 | 1000 | 4000 | 65.99 | 71.69 | 71.81 | 72.38 |

## 5 Discussions and Conclusions

Due to experiment design and computation resources, some limitations are difficult to solve in this paper. The generalization ability of the framework should be further investigated since we only use the WebNLG dataset to fine-tune the pre-trained transformers. Datasets covering other topics should be introduced to improve reasoning ability. For the text augmentation module, other models, such as `davinci`-002, are not covered, which also might lead to different performances on the generated graph. Another issue is the quality of the generated graph. Parameter settings, such as the numbers of epochs for training in graph-to-text and text-to-graph models, are fixed in our experiments, and some generated graphs cannot explicitly capture the genuine entities and relationships revealed in some complicated and long text. Thus the quality of the generated synthetic graph still has room for improvement. Evaluation of the quality of the synthetic graph is also a potential direction for future research.

To summarize this paper, we utilized the large language models to generate synthetic knowledge graph datasets. The main contributions of our work can mainly be divided into the following aspects. First, we propose a novel and straightforward framework to generate synthetic knowledge graph datasets that can be used in down-streaming tasks. This framework does not involve model architecture design and is simple to follow by leveraging the reasoning ability of large language models. Second, in our data augmentation module, we explore two different approaches, prompt and finetuning, and provide templates of design and data clean methods to generate synthetic text that can be extracted to form triples. Finally, through extensive experiments and evaluations on knowledge graph-related tasks, we show the promising of our framework for applications.

## References

Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, 2023.

Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*, 2018.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7383–7390, 2020.

Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*, 2017.

Pere-Lluís Huguet Cabot and Roberto Navigli. Rebel: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 2370–2381, 2021.

Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiao Han, Xiaohai Hu, Xuanwen Huang, and Yang Yang. Graphllm: Boosting graph reasoning ability of large language model. *arXiv preprint arXiv:2310.05845*, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. Kgpt: Knowledge-grounded pre-training for data-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8635–8648, 2020.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61, 2024.

Eli Chien, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, Jiong Zhang, Olgica Milenkovic, and Inderjit S Dhillon. Node feature extraction by self-supervised multi-scale neighborhood prediction. *arXiv preprint arXiv:2111.00064*, 2021.

Anthony Colas, Mehrdad Alvandipour, and Daisy Zhe Wang. Gap: A graph-aware language model framework for knowledge graph-to-text generation. *arXiv preprint arXiv:2204.06674*, 2022.

Claude Coulombe. Text data augmentation made simple by leveraging nlp cloud apis. *arXiv preprint arXiv:1812.04718*, 2018.

Pierre Dognin, Inkit Padhi, Igor Melnyk, and Payel Das. Regen: Reinforcement learning for text and knowledge base generation using pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1084–1099, 2021.

P ERDdS and A R&wi. On random graphs i. *Publ. math. debrecen*, 6(290-297):18, 1959.

Hamid Ghorbani. Mahalanobis distance and its application for detecting multivariate outliers. *Facta Universitatis, Series: Mathematics and Informatics*, pp. 583–595, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Aditya Grover, Aaron Zweig, and Stefano Ermon. Graphite: Iterative generative modeling of graphs. In *International conference on machine learning*, pp. 2434–2444. PMLR, 2019.

Qipeng Guo, Zhijing Jin, Xipeng Qiu, Weinan Zhang, David Wipf, and Zheng Zhang. Cycle-clegt: Unsupervised graph-to-text and text-to-graph generation via cycle training. *arXiv preprint arXiv:2006.04702*, 2020.

Zhijiang Guo, Yan Zhang, Zhiyang Teng, and Wei Lu. Densely connected graph convolutional networks for graph-to-sequence learning. *Transactions of the Association for Computational Linguistics*, 7:297–312, 2019.

Xiaoxin He, Xavier Bresson, Thomas Laurent, and Bryan Hooi. Explanations as features: Llm-based features for text-attributed graphs. *arXiv preprint arXiv:2305.19523*, 2023.

Zhiting Hu, Bowen Tan, Russ R Salakhutdinov, Tom M Mitchell, and Eric P Xing. Learning data manipulation for augmentation and weighting. *Advances in Neural Information Processing Systems*, 32, 2019.

Shima Imani, Liang Du, and Harsh Shrivastava. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*, 2023.

Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random graphs*. John Wiley & Sons, 2011.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

Mihir Kale and Abhinav Rastogi. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pp. 97–102, 2020.

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

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. Text generation from knowledge graphs with graph transformers. In *Proceedings of NAACL-HLT*, pp. 2284–2293, 2019.

Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. Kronecker graphs: an approach to modeling networks. *Journal of Machine Learning Research*, 11(2), 2010.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Yitong Li, Trevor Cohn, and Timothy Baldwin. Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 21–27, 2017.

Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. A survey of graph meets large language model: Progress and future directions. *arXiv preprint arXiv:2311.12399*, 2023.

Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Will Hamilton, David K Duvenaud, Raquel Urtasun, and Richard Zemel. Efficient graph generation with graph recurrent attention networks. *Advances in neural information processing systems*, 32, 2019.

Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. Data boost: Text data augmentation through reinforcement learning guided conditional generation. *arXiv preprint arXiv:2012.02952*, 2020.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5755–5772, 2022.

Priya Mahadevan, Dmitri Krioukov, Kevin Fall, and Amin Vahdat. Systematic topology analysis and generation using degree correlations. *ACM SIGCOMM Computer Communication Review*, 36(4):135–146, 2006.

Pedro Henrique Martins, Zita Marinho, and André FT Martins. Joint learning of named entity recognition and entity linking. *arXiv preprint arXiv:1907.08243*, 2019.

Igor Melnyk, Pierre Dognin, and Payel Das. Knowledge graph generation from text. *arXiv preprint arXiv:2211.10511*, 2022.

Alexander Podolskiy, Dmitry Lipin, Andrey Bout, Ekaterina Artemova, and Irina Piontkovskaya. Revisiting mahalanobis distance for transformer-based out-of-domain detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 13675–13682, 2021.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

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

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

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

Martin Schmitt, Leonardo FR Ribeiro, Philipp Dufter, Iryna Gurevych, and Hinrich Schütze. 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 (TextGraphs-15)*, pp. 10–21, 2021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.

Haoyue Shi, Karen Livescu, and Kevin Gimpel. Substructure substitution: Structured data augmentation for nlp. *arXiv preprint arXiv:2101.00411*, 2021.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

Dianbo Sui, Xiangrong Zeng, Yubo Chen, Kang Liu, and Jun Zhao. Joint entity and relation extraction with set prediction networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017.

Somin Wadhwa, Silvio Amir, and Byron C Wallace. Revisiting relation extraction in the era of large language models. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2023, pp. 15566. NIH Public Access, 2023.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36, 2024.

Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. Zero-shot information extraction via chatting with chatgpt. *arXiv preprint arXiv:2302.10205*, 2023.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33:6256–6268, 2020.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. A partition filter network for joint entity and relation extraction. *arXiv preprint arXiv:2108.12202*, 2021.

Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. Graphformers: Gnn-nested transformers for representation learning on textual graph. *Advances in Neural Information Processing Systems*, 34:28798–28810, 2021.

Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International conference on machine learning*, pp. 5708–5717. PMLR, 2018.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers*, pp. 2335–2344, 2014.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. Position-aware attention and supervised data improve slot filling. In *Conference on empirical methods in natural language processing*, 2017.

Jianan Zhao, Le Zhuo, Yikang Shen, Meng Qu, Kai Liu, Michael Bronstein, Zhaocheng Zhu, and Jian Tang. Graphtext: Graph reasoning in text space. *arXiv preprint arXiv:2310.01089*, 2023.

# A Appendix

## A.1 Training Details

In this paper, we trained graph-to-text and text-to-graph modules locally, and fine-tuned `gpt-3.5-turbo` with OpenAI API. The source codes are from the following links, and we also modified part of the codes due to the error caused by old versions of Pytorch and transformers. Table 4 to 7 provide details for some important hyperparameters we used in our experiments.

**GAP**Colas et al. (2022):https://github.com/acolas1/GAP_COLING2022

**Grapher**Melnyk et al. (2022):https://github.com/IBM/Grapher

**PFN**Yan et al. (2021):https://github.com/Coopercoppers/PFN

**SPN**Sui et al. (2023):https://github.com/DianboWork/SPN4RE

Table 4: Hyperparameter Details for GAP

| Hyperparameter Name | Settings | Hyperparameter Name | Settings |
|---|---|---|---|
| number of beams | 5 | max node length | 50 |
| train batch size | 16 | predict batch size | 16 |
| max input length | 256 | max output length | 128 |
| learning rate | 2e-5 | number of train epochs | 40 |
| warmup steps | 1600 | eval period | 500 |

Table 5: Hyperparameter Details for Grapher

| Hyperparameter Name | Settings | Hyperparameter Name | Settings |
|---|---|---|---|
| number of epochs | 50 | number of nodes | 1 |
| edges as classes | 1 | learning rate | 1e-4 |
| batch size | 11 | focal loss gamma | 3 |
| dropout rate | 0.5 | number of layers | 2 |

Table 6: Hyperparameter Details for PFN

| Hyperparameter Name | Settings | Hyperparameter Name | Settings |
|---|---|---|---|
| number of epochs | 20 | batch size | 20 |
| learning rate | 2e-5 | evaluation metric | macro F1 |
| hidden size | 300 | clip rate | 0.25 |
| dropout rate | 0.1 | max sequence length | 128 |

Table 7: Hyperparameter Details for SPN

| Hyperparameter Name | Settings | Hyperparameter Name | Settings |
|---|---|---|---|
| number of epochs | 50 | batch size | 4 |
| learning rate of decoder | 5e-5 | learning rate of encoder | 2e-5 |
| learning rate decay | 0.02 | max span length | 10 |
| head entity loss weight | 2 | tail entity loss weight | 2 |
| relation loss weight | 1 | optimizer | AdamW |

### A.2 GENERATED SAMPLES

In this section, we provided some samples of the text augmentation procedure. Section A.2.1 shows the input and initial output by the prompt approach, and Section A.2.2 provides some fine-tuning data samples and initial output by the fine-tuning approach. We use the word "initial" here since they are not separated, cleaned and selected by the calculated distance.

#### A.2.1 INPUT AND INITIAL OUTPUT BY PROMPT

**Input:**

Given some sentences that describe some facts as follows, generate other twenty sentences with similar language complexity and sentence length range to them that describe other facts:

Sentence 1. Agustin Barboza was born in Asuncion, Paraguay, where the leader is Juan Afara.

Sentence 2. Cake is a dessert, as is Bionico.

Sentence 3. American Buzz Aldrin graduated from the Massachusetts Institute of Technology in 1963 with a Sc. D.. and was selected by NASA in 1963.

Sentence 4. Alberto Teisaire, who worked as the Provisional President of the Argentine Senate, was succeeded by Isaac Rojas.

Sentence 5.103 Hera was discovered on July 7th, 1868 by James Craig Watson, a Canadian national. His alma mater was the University of Michigan and he died in Madison, Wisconsin.

Sentence 6.John Buscema won the Eisner Award .

Sentence 7. Dunwoody is the largest city in DeKalb County, Georgia.

Sentence 8. The AWH Engineering College in Kerala, India has 250 academic staff.

Sentence 9. Adare Manor is located in County Limerick which is part of Munster.

Sentence 10. American William Anders was born in British Hong Kong. He was a crew member of NASA's Apollo 8 along with Frank Borman and Buzz Aldrin as backup pilot. He retired on September 1st, 1969.

Sentence 11.Ace Wilder was born in Sweden in 1982. He is a solo singer and songwriter.

Sentence 12. The comic character Blockbuster, also known as Mark Desmond, was created by Carmine Infantino and Gardner Fox.

Sentence 13. Buzz Aldrin was selected by NASA in 1963.

Sentence 14. The runway length of Andrews County Airport is 896.0.

Sentence 15. The largest city in Oregon is Portland.

Sentence 16. Anders Osborne's genre is Rhythm and blues.

Sentence 17. Christian Panucci plays for Inter Milan.

Sentence 18.Christian Panucci plays for Chelsea F.C .

Sentence 19. Alan Shepard was born in New Hampshire and died in California. He was awarded the Distinguished Service Medal by the United States Navy which ranks higher than the Department of Commerce Gold Medal.

Sentence 20. Aaron Hunt played for SV Werder Bremen II and now plays for VfL Wolfsburg, which is managed by Dieter Hecking .

**Initial Output:**

Sentence 1. Maria Lopez was born in Madrid, Spain, where the mayor is Manuel Garcia.

Sentence 2. Ice cream is a popular dessert, as are macarons.

Sentence 3. English geneticist Gregor Mendel graduated from the University of Hohenheim in 1853 with a Ph.D. and conducted his famous pea plant experiments in the 1860s.

Sentence 4. Juan Martinez, who served as the Chief Justice of the Supreme Court in Peru, was succeeded by Maria Ramirez.

Sentence 5. Asteroid 209 Dido was discovered on March 6th, 1879, by Johann Palisa, an Austrian astronomer. He studied at the University of Vienna and passed away in Graz, Austria.

Sentence 6. Frank Miller won the Harvey Award.

Sentence 7. Marietta is the largest city in Cobb County, Georgia.

Sentence 8. The University of Engineering and Technology in Lima, Peru, has 300 academic staff members.

Sentence 9. Adare Castle is located in County Tipperary, which is part of Munster.

Sentence 10. Canadian astronaut Chris Hadfield was born in Sarnia, Ontario, and was the first Canadian to walk in space during Mission STS-74. He retired from the Canadian Space Agency in 2013.

Sentence 11. Sofia Lindgren was born in Norway in 1990. She is a professional dancer and choreographer.

Sentence 12. The superhero character Nightwing, also known as Dick Grayson, was created by Bob Kane and Bill Finger.

Sentence 13. Buzz Aldrin was initially chosen by NASA in 1963.

Sentence 14. The runway length of London Luton Airport is 878.0 meters.

Sentence 15. The capital of South Dakota is Pierre.

Sentence 16. John Smith's genre is Country music.

Sentence 17. Cristiano Ronaldo plays for Real Madrid.

Sentence 18. Cristiano Ronaldo plays for Juventus F.C.

Sentence 19. Ellen Ochoa was born in California and died in Arizona. She was the first Hispanic woman to go to space.

Sentence 20. Marco Reus played for Borussia Mönchengladbach and now plays for Borussia Dortmund, which is managed by Lucien Favre.

### A.2.2 INPUT AND INITIAL OUTPUT BY FINETUNING

The following provides two input samples for fine-tuning and two output samples when $\alpha = 5$. All input samples are stored in a `jsonl` file. For different values of $\alpha$, the only difference is the number of sentences in the assistant's content.

**Input Sample 1**: {"messages": [{"role": "system", "content": "This is a factual and objective chatbot."}, {"role": "user", "content": "Generate some sentences that describe facts."}, {"role": "assistant", "content": " Sentence 1. Allama Iqbal International Airport is located in Pakistan and operated by the Pakistan Civil Aviation Authority. The airport serves the city of Lahore and has a runway length of 2900.0. Sentence 2. Bakewell pudding is a dessert from the Derbyshire Dales region. The main ingredients are ground almonds, jam, butter and eggs. Sentence 3. The United States House of Representatives is located in the U.S. and leads Anaheim, California. Sentence 4. Beef kway teow is a popular dish in Singapore and Indonesia. Oyster sauce is an ingredient in Beef Kway Teow. Sentence 5. Athens International Airport serves the city of Athens in Greece where the Greek language is spoken. Two of the leaders of the country are Alexis Tsipras and Prokopis Pavlopoulos ."}]}

**Input Sample 2**: {"messages": [{"role": "system", "content": "This is a factual and objective chatbot."}, {"role": "user", "content": "Generate some sentences that describe facts."}, {"role": "assistant", "content": " Sentence 1. Ashgabat International Airport is 211 metres above sea level and its 1st runway is 12467 feet in length. Sentence 2. Rock and roll originated from blues music. Sentence 3. Andrew White started his musical career in 2003 and is associated with the musical artist Marry Banilow. He is signed to the now-defunct Universal Records label and plays for the band Kaiser Chiefs. Sentence 4. Andra is a rhythm and blues singer. Rhythm and blues originated from blues music and disco and funk are derivatives of it. Sentence 5. The main ingredients of Batagor are fried fish dumplings with tofu and vegetables in peanut sauce ."}]}

**Output Sample 1**:

Sentence 1. Bread is an ingredient of a bacon sandwich.

Sentence 2. Parachute Publishing was founded in 1986.

Sentence 3. The Aston Martin V8 is produced by Aston Martin.

Sentence 4. Bakewell tart is a variation of Bakewell pudding which has almonds as an ingredient.

Sentence 5. Bibite is an Italian word which comes from the plural of the Latin word bibo. The singular word is bibo.

**Output Sample 2**:

Sentence 1. Addison County Field Days takes place in Addison, Vermont.

Sentence 2. Asam pedas is a food from Malaysia.

Sentence 3. A Fortress of Grey Ice is from the United States where the capital city is Washington D.C . and Asian Americans are an ethnic group.

Sentence 4. The 1st School of Law in Si Sa Ket was established in 2006 and has 3500 students. It is affiliated with Rajabhat Suan Dusit University.

Sentence 5. Adolfo Suárez Madrid–Barajas Airport is operated by ENAIRE where Benito Nardone is the headquarters.

## A.3 BROADER IMPACT

The proposed framework can contribute to augmenting knowledge graph datasets but also other types of graphs with textual attributes. The efficiency of our method depends on the text augmentation module design. If the fine-tuned model possesses knowledge from different domains, it might be capable of generating graph datasets with different topics. The performance of models related to graphs can also potentially benefit from the increasing size of training data, such as graph neural networks.