Fine-Tuning vs. Prompting: Evaluating the Knowledge Graph Construction with LLMs

Hussam Ghanem[1,2], Christophe Cruz[1,*,†]

[1]ICB, UMR 6306, CNRS, Université de Bourgogne, 21000 Dijon, France
[2]DAVI The Humanizers, Puteaux, France

Abstract

This paper explores Text-to-Knowledge Graph (T2KG) construction„ assessing Zero-Shot Prompting (ZSP), Few-Shot Prompting (FSP), and Fine-Tuning (FT) methods with Large Language Models (LLMs). Through comprehensive experimentation with Llama2, Mistral, and Starling, we highlight the strengths of FT, emphasize dataset size's role, and introduce nuanced evaluation metrics. Promising perspectives include synonym-aware metric refinement, and data augmentation with LLMs. The study contributes valuable insights to KG construction methodologies, setting the stage for further advancements.[1]

Keywords

Text-to-Knowledge Graph, Large Language Models, Zero-Shot Prompting, Few-Shot Prompting, Fine-Tuning

1. Introduction

The term "knowledge graph" has been around since 1972, but its current definition can be traced back to Google in 2012. This was followed by similar announcements from companies such as Airbnb, Amazon, eBay, Facebook, IBM, LinkedIn, Microsoft, and Uber, among others, leading to an increase in the adoption of Knowledge graphs(KGs) by various industries. As a result, academic research in this field has seen a surge in recent years, with an increasing number of scientific publications on KGs [1]. These graphs utilize a graph-based data model to effectively manage, integrate, and extract valuable insights from large and diverse datasets [2].

KGs serve as repositories for structured knowledge, organized into a collection of triples, denoted as $KG = (h, r, t) \subseteq E \times R \times E$, where E represents the set of entities, and R represents the set of relations [1]. Within a graph, nodes represent various levels, entities, or concepts. These nodes encompass diverse types, including person, book, or city, and are interconnected by relationships such as located in, lives in, or works with. The essence of a KG emerges when it incorporates multiple types of relationships rather than being confined to a single type. The overarching structure of a KG constitutes a network of entities, featuring their semantic types, properties, and interconnections. Thus, constructing a KG necessitates information about

[1]Our code at https://github.com/ChristopheCruz/LLM4KGC

3rd International Workshop on Knowledge Graph Generation from Text (Text2KG), Co-located with the Extended Semantic Web Conference (ESWC 2024) , May 26–30, 2024, Hersonissos, Greece

*Corresponding author.
†These authors contributed equally.
hussam.ghanem@u-bourgogne.fr (H. Ghanem); christophe.cruz@u-bourgogne.fr (C. Cruz)
https://github.com/ChristopheCruz/LLM4KGC (C. Cruz)

entities (along with their types and properties) and the semantic relationships that bind them. For the extraction of entities and relationships, practitioners often turn to NLP tasks like Named Entity Recognition (NER), Coreference Resolution (CR), and Relation Extraction (RE).

KGs play a crucial role in organizing complex information across diverse domains, such as question answering, recommendations, semantic search, etc. However, the ongoing challenge persists in constructing them, particularly as the primary sources of knowledge are embedded in unstructured textual data such as press articles, emails, and scientific journals. This challenge can be addressed by adopting an information extraction approach, sometimes implemented as a pipeline. It involves taking textual inputs, processing them using Natural Language Processing (NLP) techniques, and leveraging the acquired knowledge to construct or enhance the KG.

If we envision the Text-to-Knowledge Graph (T2KG) construction task as a black box, the input is textual data, and the output is a knowledge graph. Achieving this can be approached through methods that directly convert text into a graph or by implementing NLP tasks in two ways: 1) through an information extraction pipeline incorporating the mentioned tasks independently, or 2) by adopting an end-to-end approach, also known as joint prediction, using Large Language Models (LLMs) for example. In the realm of LLMs and KGs, their mutual enhancement is evident. LLMs can assist in the construction of KGs. Conversely, KGs can be employed to validate outputs from LLMs or provide explanations for them [3]. LLMs can be adapted to KG construction task (T2KG) through various approaches, such as fine-tuning [4] (FT), zero-shot prompting [5] (ZSP), or few-shot prompting (FSP) [6] with a limited number of examples. Each of these approaches has their pros and cons with respect to the performance, computation resources, training time, domain adaption and training data required.

In-context learning, as discussed by [7], coupled with prompt design, involves telling a model to execute a new task by presenting it with only a few demonstrations of input-output pairs during inference. Instruction fine-tuning methods, exemplified by InstructGPT [8] and Reinforcement Learning from Human Feedback (RLHF) [9], markedly enhance the model's ability to comprehend and follow a diverse range of written instructions. Numerous LLMs have been introduced in the last year, as highlighted by [3], particularly within the ChatGPT [10] like models, which includes GPT-3 [11], LLaMA [12], BLOOM [13], PaLM [14], Mistral [15], Starling [16] and Zephyr [17]. These models can be readily repurposed for KG construction from text by employing a prompt design that incorporates instructions and contextual information.

This study does not entail a comparison with traditional methods of constructing KGs; rather, it delves into the developments and challenges associated with KG construction methodologies, and aiming at providing formal evaluation of T2KG task. Specifically, we focus on the utilization of LLMs, and explore the three approaches mentioned before, Zero-shot, Few-shot and Fine-tuning (Fig. 1). Each of these approaches addresses specific challenges, contributing significantly to the evolution of T2KG construction techniques.

The present study is organized as follows, Section 2 presents a comprehensive overview of the current state-of-the-art approaches for Text to KG (T2KG) Construction. In the Section 3, we present the general architecture of our proposed implementation (method), with datasets, metrics, and experiments. Section 4 then encapsulates the findings and discussions, presenting the culmination of results. Finally, Section 5 critically examines the strengths and limitations of these techniques.

2. Background

The current state of research on knowledge graph construction using LLMs is discussed. Three main approaches are identified: Zero-Shot, Few-Shot, and Fine-Tuning. Each approach has its own challenges, such as maintaining accuracy without specific training data or ensuring the robustness of models in diverse real-world scenarios. Evaluation metrics used to assess the quality of constructed KGs are also discussed, including semantic consistency and linguistic coherence. This section highlight methods and metrics to construct KGs and evaluate the result.

The figure 1 illustrates the black box joint prediction of the T2KG construction process using LLMs. It demonstrates how two French examples on the left are converted into an expected result (KG) on the right using ZSP, FSP or FT approaches with LLMs.

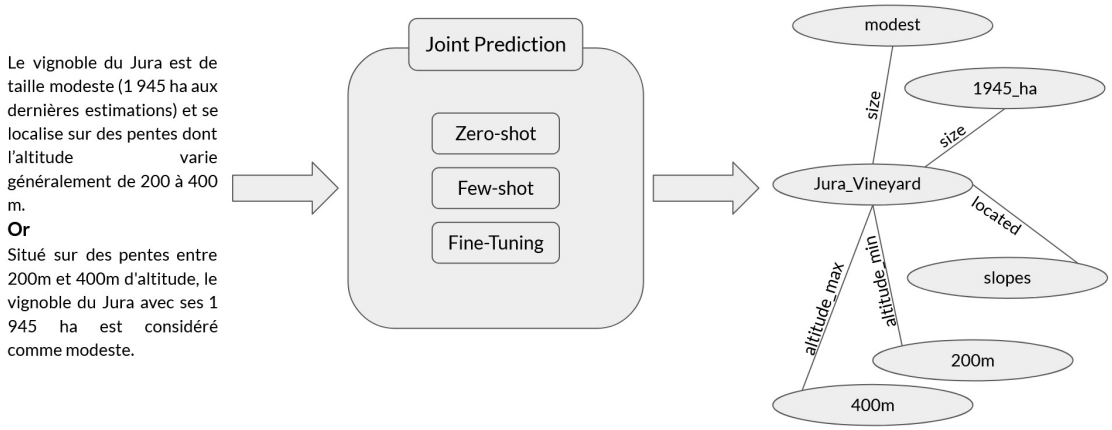

Figure 1: T2KG Task

2.1. Zero Shot

Zero Shot methods enable KG construction without task-specific training data, leveraging the inherent capabilities of large language models. [18] introduce an innovative approach using large language models (LLMs) for knowledge graph construction, employing iterative zero-shot prompting for scalable and flexible KG construction. [19] evaluate the performance of LLMs, specifically GPT-4 and ChatGPT, in KG construction and reasoning tasks, introducing the Virtual Knowledge Extraction task and the VINE dataset, but they do not take into account open sourced LLMs as LLaMA [12]. [20] assess ChatGPT's abilities in information extraction tasks, identifying overconfidence as an issue and releasing annotated datasets. [21] tackle zero-shot information extraction using ChatGPT, achieving impressive results in entity relation triple extraction. [22] propose a method for Knowledge Graph Construction (KGC) using an analogy-based approach, demonstrating superior performance on Wikidata. [23] address the limitations of existing generative knowledge graph construction methods by leveraging large generative language models trained on structured data. The most of these approaches having the same limitation, which is the use of closed and huge LLMs as ChatGPT or GPT4 for this

task. Challenges in this area include maintaining accuracy without specific training data and addressing nuanced relationships between entities in untrained domains.

2.2. Few Shot

Few Shot methods focus on constructing KGs with limited training examples, aiming to achieve accurate knowledge representation with minimal data. [6] introduce PiVe, a framework enhancing the graph-based generative capabilities of LLMs, and the authors create a verifier which is responsable to verifie the results of LLMs with multi-iteration type. [24] explore the potential of LLMs for knowledge graph completion, treating triples as text sequences and utilizing LLM responses for predictions. [25] automate the process of generating structured knowledge graphs from natural language text using foundation models. [26] present OpenBG, an open business knowledge graph derived from Alibaba Group, containing 2.6 billion triples with over 88 million entities. [27] explore the integration of LLMs with semantic technologies for reasoning and inference. [28] investigate LLMs' application in relation labeling for e-commerce Knowledge Graphs (KGs). As ZSP approaches, FSP approaches use closed and huge LLMs as ChatGPT or GPT4 [10] for this task. Challenges in this area include achieving high accuracy with minimal training data and ensuring the robustness of models in diverse real-world scenarios.

2.3. Fine-Tuning

Fine-Tuning methods involve adapting pre-trained language models to specific knowledge domains, enhancing their capabilities for constructing KGs tailored to particular contexts. [4] present a case study automating KG construction for compliance using BERT-based models. This study emphasizes the importance of machine learning models in interpreting rules for compliance automation. [29] propose an approach for knowledge extraction and analysis from biomedical clinical notes, utilizing the BERT model and a Conditional Random Field layer, showcasing the effectiveness of leveraging BERT models for structured biomedical knowledge graphs. [30] propose Knowledge Graph-Enhanced Large Language Models (KGLLMs), enhancing LLMs with KGs for improved factual reasoning capabilities. These approaches that applied FT, they do not use new generations of LLMs, specially, decoder only LLMs as Llama, and Mistral. Challenges in this domain include ensuring the scalability, interpretability, and robustness of fine-tuned models across diverse knowledge domains.

2.4. Evaluation metrics

As we employ LLMs to construct KGs, and given that LLMs function as Natural Language Generation (NLG) models, it becomes imperative to discuss NLG criteria. In NLG, two criteria [31] are used to assess the quality of the produced answers (triples in our context).

The first criterion is semantic consistency or Semantic Fidelity which quantifies the fidelity of the data produced against the input data. The most common indicators are :

- **Hallucination**: It is manifested by the Presence of information (facts) in the generated text that is absent in the input data. In our scenario, hallucination exists if the generated triples (GT) contain triples not present in the ground truth triples (ET) (T in GT and not

in ET);

- **Omission**: It is manifested by the omission of one of the pieces of information (facts) in the generated text. In our case, omission occurs if a triple is present in ET but not in GT;

- **Redundancy**: This is manifested by the repetition of information in the generated text. In our case, the redundancy exists if a triple appears more than once in GT;

- **Accuracy**: The lack of accuracy is manifested by the modification of information such as the inversion of the subject and the direct object complement in the generated text. Accuracy increases if there is an exact match between ET and GT. ;

- **Ordering**: It occurs when the sequence of information is different from the input data. In our case, the ordering of GT is not considered.

The second criterion is linguistic coherence or Output Fluency to evaluate the fluidity of the text and the linguistic constructions of the generated text, the segmentation of the text into different sentences, the use of anaphoric pronouns to reference entities and to have linguistically correct sentences. However, in our evaluation, we do not take into account the second criterion.

In their experiments, [3] calculated three hallucination metrics - subject hallucination, relation hallucination, and object hallucination - using certain preprocessing steps such as stemming. They used the ground truth ontology alongside the ground truth test sentence to determine if an entity or relation is present in the text. However, a limitation could arise when there is a disparity in entities or relations between the ground truth ontology and the ground truth test sentence. If the generated triples contain entities or relations not present in the ground truth text, even if they exist in the ground truth ontology, it will be considered a hallucination.

The authors of [6] evaluate their experiments using several evaluation metrics, including Triple Match F1 (T-F1), Graph Match F1 (G-F1), G-BERTScore (G-BS) from [32] which extends BertScore [33] for graph matching, and Graph Edit Distance (GED) from [34]. The GED metric measures the distance between the predicted graph and the ground-truth graph, which is equivalent to computing the number of edit operations (addition, deletion, or replacement of nodes and edges) needed to transform the predicted graph into a graph that is identical to the ground-truth graph, but it does not provide a specific path for these operations to calculate the exact number of operations. To adhere with semantic consistency criterion, we use the terms "omission" and "hallucination" in place of "addition" and "deletion," respectively.

3. Propositions

This section describes our approach to evaluate the quality of generated KGs. We explain how we use evaluation metrics such as T-F1, G-F1, G-BS, GED, Bleu-F1 [35] and ROUGE-F1 [36] to assess the quality of the generated KGs in comparison to ground-truth KGs. Additionally, we discuss the use of Optimal Edit Paths (OEP) metric [1] to determine the precise number of

[1]NetworkX - optimal edit paths : https://networkx.org/documentation/stable/index.html

operations required to transform the predicted graph into an identical representation of the ground-truth graph. This metric serves as a basis for calculating omissions and hallucinations in the generated graphs. We employ examples from the WebNLG+2020 dataset [37] for testing with ZSP and FSP techniques. Additionally, we utilize the training dataset of WebNLG+2020 to train LLMs using the FT technique. Subsequent subsections delve into a detailed discussion of each phase.

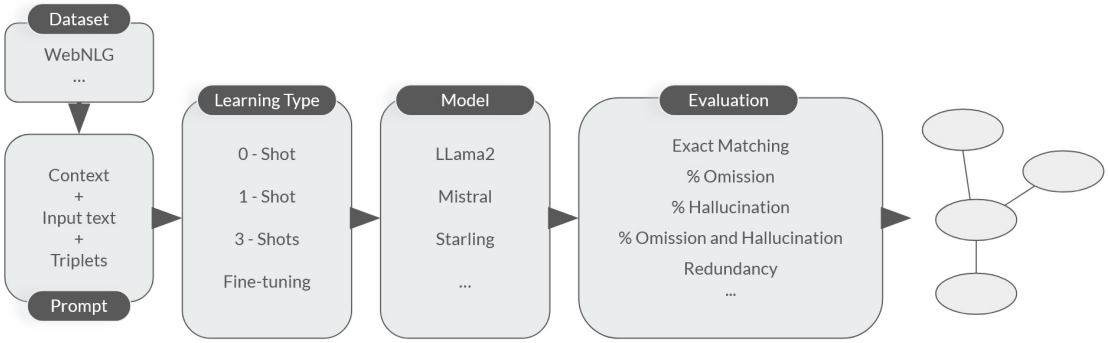

Figure 2: Overall experimentation's process

3.1. Overall experimentation's process

We leverage the WebNLG+2020 dataset, specifically the version curated by [6]. Their preparation of graphs in lists of triples proves beneficial for evaluation purposes. We utilize these lists and employ NetworkX [38] to transform them back into graphs, facilitating evaluations on the resultant graphs. This step is instrumental in performing ZSP, FSP, and FT LLMs on this dataset.

The figure 2 illustrates the different stages of our experimentation process, including data preparation, model selection, training, validation, and evaluation. The process begins with data preparation, where the WEBNLG dataset is preprocessed and split into training, validation, and test sets. Next, the learning type is selected, and different models are trained using the training set. The trained models are then evaluated on the validation set to evaluate their performance. Finally, the best-performing model is selected and validated on the test set to estimate its generalization ability.

3.2. Prompting learning

During this phase, we employ the ZSP and FSP techniques on LLMs to evaluate their proficiency in extracting triples (e.g. construction of the KG). The application of these techniques involves merging examples from the test dataset of WebNLG+2020 with our adapted prompt. Our prompt is strategically modified to provide contextual guidance to the LLMs, facilitating the effective extraction of triples, without the inclusion of a support ontology description, as demonstrated in [3]. The specific prompts used for ZSP and FSP are illustrated in Fig 3(a) and Fig 3(b),

In our approach for ZSP, we began with the methodology outlined in [6], initiating our prompt with the directive "Transform the text into a semantic graph." However, we enhanced

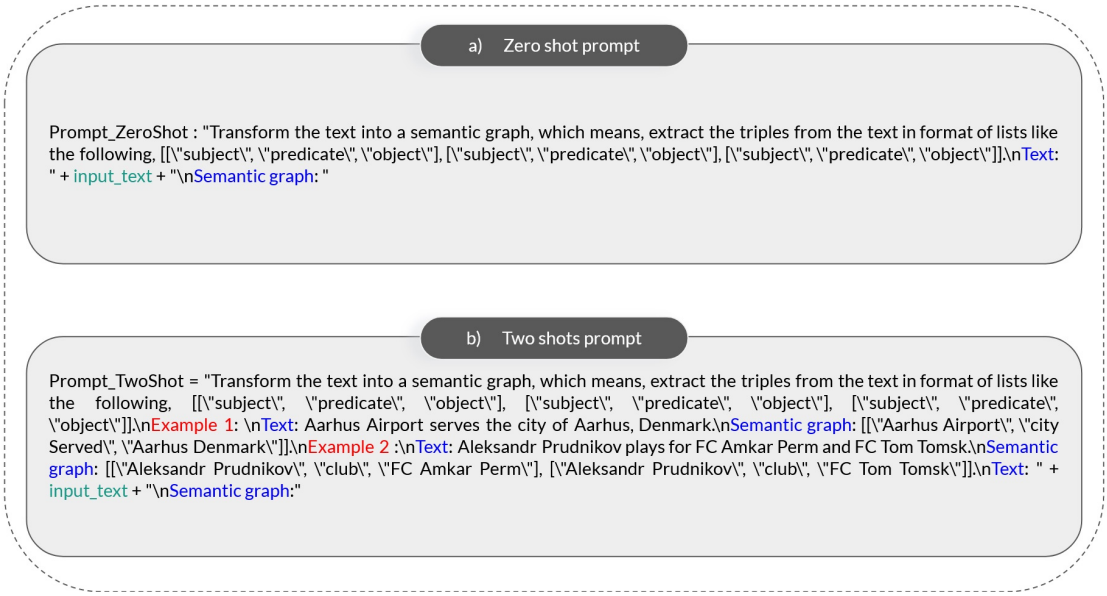

Figure 3: Prompting examples

this prompt by incorporating additional sentences tailored for our LLMs, as illustrated in Fig 3.(a).

For FSP, we executed 7-shots learning. The rationale behind employing 7-shots learning lies in the fact that the maximum KG size in WebNLG+2020 is 7 triples. Consequently, we fed our prompt with 7 examples of varying sizes; example 1 with size 1, example 2 with size 2, example 3 with size 3, and so forth. In Figure 3-b, we depict a prompt containing two examples.

To demonstrate the efficacy of our refined prompt (including additional sentences), we conducted zero-shot experiments on ChatGPT [10], comparing the outcomes with those of [6]. Our results consistently reveal that our prompt yields more coherent answers in terms of structure.

3.3. Finetuning

If the initial results from the ZSP and FSP on LLMs prove reasonable, we proceed to the FT phase. This phase aims to provide the LLMs with a more specific context and knowledge related to the task of extracting triples within the domains covered by the WebNLG+2020 dataset. Using the example "a)" illustrated in Fig 3, we passe in the FT prompt, at once for each line of the training dataset, the input text and the corresponding KG (the list of triples). To do this phase (FT), we employ QLoRA [39], a methodology that integrates quantization [40] and Low-Rank Adapters (LoRA) [41]. The LLM is loaded with 4-bit precision using bitsandbytes [42], and the training process incorporates LoRA through the PEFT library (Parameter-Efficient Fine-Tuning) [43] provided by Hugging Face.

3.4. Postprocessing

Given our focus on KG construction, our evaluation process involves assessing the generated KGs against ground-truth KGs. To facilitate this evaluation, we take a cleaning process for the LLMs output. This involves transforming the graphs generated by LLMs into organized lists of triples, subsequently transferred to textual documents.

The transformation is executed through rule-based processing. This step is applied to remove corrupted text (outside the lists of triples) from the whole text generated by LLMs in the preceding step. The output is then presented in a list of lists of triples format, optimizing our evaluation process. This approach proves especially effective when calculating metrics such as G-F1, GED, and OEP, as we will see in more detail in 3.5

A potential problem arises when instructing LLMs to produce lists of triples (KGs), as there may be instances where the generated text lacks the desired structure. In such cases, we address this issue by substituting the generated text with an empty list of triples, represented as '[["","","",""]]', allowing us to effectively evaluate omissions. However, this approach tends to underestimate hallucinations compared to the actual occurrences.

3.5. Experiment's evaluation

For assessing the quality of the generated graphs in comparison to ground-truth graphs, we adopt evaluation metrics as employed in [6]. These metrics encompass T-F1, G-F1, G-BS [32], and GED [34]. Additionally, we incorporate the Optimal Edit Paths (OEP) metric, a tool aiding in the calculation of omissions and hallucinations within the generated graphs.

Our evaluation procedure aligns with the methodology outlined in [6], particularly in the computation of GED and G-F1. This involves constructing directed graphs from lists of triples, referred to as linearized graphs, utilizing NetworkX [38].

In contrast to [3], our methodology diverges by not relying on the ground truth test sentence of an ontology. As previously mentioned, we opt for a distinct approach wherein we assess omissions and hallucinations in the generated graphs using the OEP metric. Unlike the global edit distance provided by GED, OEP gives the precise path of the edit, enabling the exact quantification of omissions and hallucinations, either in absolute terms or as a percentage across the entire test dataset.

For example, in the illustrated nodes path labeled 'a)' in Fig 4-(b), we observe 2 omissions, while the edges path in Fig 4-(a) exhibits 1 hallucination. In our evaluation, the criterion for incrementing the global hallucination metric for all graphs is set at finding >=1 hallucinations or 1 omission in a generated graph. This approach ensures a comprehensive assessment of the presence of omissions and hallucinations across the entirety of the generated graphs.

As mentioned earlier, the evaluation of the three methods is conducted using examples sourced from the test dataset of WebNLG+2020. The primary goal is to enhance G-F1, T-F1, G-BS, Bleu-F1, and ROUGE-F1 metrics, while reducing GED, Hallucination, and Omission.

3.6. Mathematical representation of the used metrics

We mathematically represent the used metrics as follows:

Graph Matching (G-$F1$). Let Mch be the number of matches between predicted and gold graphs. And let $ToGraphs$ be the total number of predicted graphs. Then, the accuracy for entire graph matches Acc_{graph} can be calculated as:

$$ACC_{graph} = \frac{Mch}{ToGraphs}$$

Triples Matcning (T-$F1$). The $F1$ score for triple matches T-$F1$ is calculated in the following:

$$T\text{-}F1 = \frac{2 \times TP}{2 \times TP + FP + FN}$$

Where

- TP is the number of true positive triple matches.
- FP is the number of false positive triple matches.
- FN is the number of false negative triple matches.

Graph Edit Distance (GED). The following equation calculate GED between two given graphs :

$$GED(g1, g2) = \min_{e_1,\ldots,e_k \in \gamma(g1,g2)} \sum_{i=1}^{k} c(e_i)$$

Where:

- GED(g_1, g_2): This represents the graph edit distance between two graphs g_1 and g_2.
- $\min_{e_1,\ldots,e_k \in \gamma(g_1,g_2)}$: This part denotes taking the minimum over all possible edit paths $e_1, \ldots, e_k$ in the set $\gamma(g_1, g_2)$. The set $\gamma(g_1, g_2)$ contains all possible edit paths that transform g_1 into g_2.
- $\sum_{i=1}^{k} c(e_i)$: This part calculates the sum of the costs of each individual edit operation e_i in the selected edit path. The cost function $c(e_i)$ measures the cost or strength of each edit operation. The objective is to find the edit path with the minimum total cost, which represents the least amount of transformation required to convert g_1 into g_2.

In our experiments, we calculate the **overall** GED which is computed as follows:

$$\text{overall\_ged} = \frac{1}{N} \sum_{i=1}^{N} \text{GED}_{\text{ED}_i}$$

Where:

- N is the total number of graphs.
- GED_{ED_i} is the graph edit distance for the ith graph.

Graph BERTScore (G-BS) . G-BS takes graphs as a set of edges and solve a matching problem which finds the best alignment between the edges in predicted graph and those in ground-truth graph. Each edge is considered as a sentence and BERTScore is used to calculate the score between a pair of predicted and ground-truth edges, Based on the best alignment and the overall matching score, the computed F1 score is used as the final G-BERTScore. Considering x_i as reference token (entity or relation) and $\hat{x}_i$ as generated token (entity or relation), the complete score matches each token in x to a generated token in $\hat{x}$ to compute recall, and each token in $\hat{x}$ to a token in x to compute precision. A greedy matching is used to maximize the matching similarity score, where each token is matched to the most similar token in the other graph. Then precision and recall are combined to compute an F1 measure. For a reference x and candidate $\hat{x}$, the recall, precision, and F1 scores are:

$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} x_i^T \hat{x}_j,$$

$$P_{\text{BERT}} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} x_i^T \hat{x}_j,$$

$$F1_{\text{BERT}} = \frac{2 \cdot P_{\text{BERT}} \cdot R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}}.$$

Bleu-F1 Score ($F1_{Bleu}$). Let C_{gen} be the count of 4-grams in the generated graph , Let C_{ref} be the count of 4-grams in the reference graph, and Let C_{match} be the count of matching 4-grams in both texts

$$P_{Bleu} = \frac{C_{match}}{C_{gen}}$$

$$R_{Bleu} = \frac{C_{match}}{C_{ref}}$$

$$F1_{Bleu} = \frac{2 \times P_{Bleu} \times R_{Bleu}}{P_{Bleu} + R_{Bleu}}$$

ROUGE-F1 Score ($F1_{ROUGE}$). In our experiments, we calculate F1-score for Rouge-2 (bi-gram), which is presented in the following equation:

$$P_{ROUGE} = \frac{bigram_{cand.} \cap bigram_{ref.}}{bigram_{cand.}}$$

$$R_{ROUGE} = \frac{bigram_{cand.} \cap bigram_{ref.}}{bigram_{ref.}}$$

$$F1_{ROUGE} = 2. \frac{R_{ROUGE}.P_{ROUGE}}{R_{ROUGE} + P_{ROUGE}}$$

Hallucination and Omission. As mentioned before, we calculate hallucination and omission using OEP, which is the optimal edit paths between the gold and predicted graphs. Each edit operation (ei) in OEP represents an action required to transform the predicted graph into the gold graph.

- **Hallucination**: An edit operation e_i is considered a hallucination if it involves adding an entity or a relation that is not present in the gold graph but exists in the predicted graph. In our work, we take into account the overall hallucination $hall.$, this metric is represented by the following equation :

$$Hall. = \frac{hall}{ToGrs}$$

  Where $hall$ is the number of graphs with hallucination, and $ToGrs$ in the total number of generated graphs
- **Omission**: An edit operation ei is considered an omission if it involves deleting an entity or a relation that exists in the gold graph but is missing in the predicted graph. In ou work, we do the same as the hallucicnation, we calculate the overall omission $omis.$, presented by the following equation :

$$Omis. = omiss/ToGrs$$

  Where $omiss$ is the number of graphs with omission.

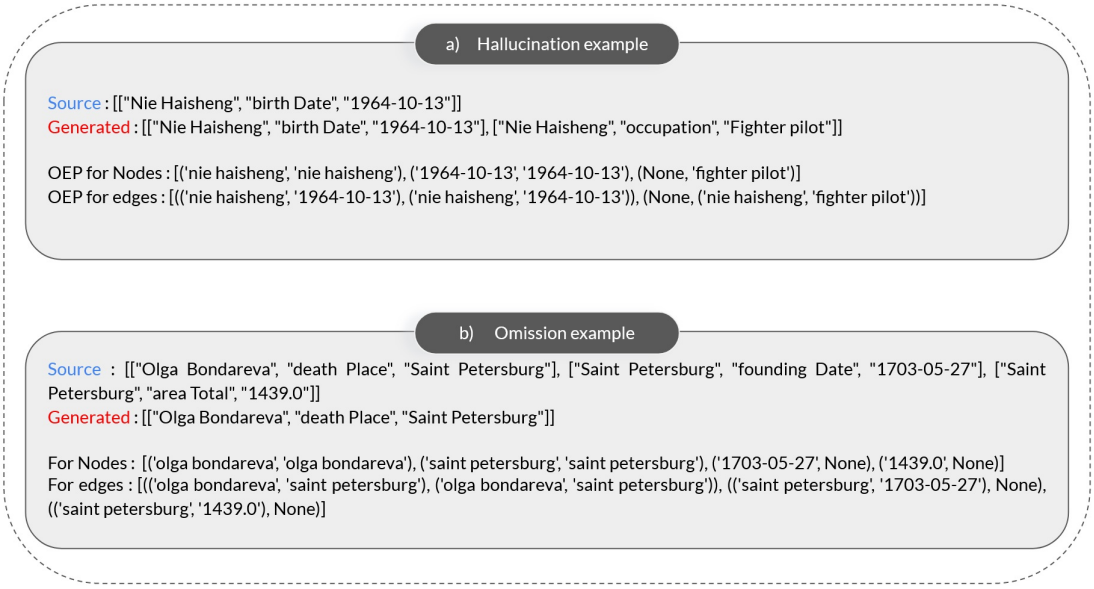

Figure 4: Results examples

4. Experiments

This section provides insights into the LLMs utilized in our study for ZSP, FSP, or FT, followed by the presentation of our experimental results.

In this section, we provide a brief overview of the LLMs utilized in our experiments. Our selection criteria focused on employing small, open-source, and easily accessible LLMs. All models were sourced from the HuggingFace platform[2]

- **Llama 2** [12] is a collection of pretrained and fine-tuned generative text models ranging in scale from 7 billion to 70 billion parameters. In our experiments, we deploy the 7B and 13B pretrained models, which have been converted to the Hugging Face Transformers format.
- Introduced by [15], **Mistral-7B-v0.1** is a pretrained generative text model featuring 7 billion parameters. Notably, Mistral-7B-v0.1 exhibits superior performance to Llama 2 13B across all benchmark tests in their experiments.
- In the work presented by [16], **Starling-7B** is introduced as an open LLM trained through Reinforcement Learning from AI Feedback (RLAIF). This model leverages the GPT-4 labeled ranking dataset, berkeley-nest/Nectar, and employs a novel reward training and policy tuning pipeline.

In our review of the state-of-the-art, we observed that, apart from [3], which incorporates hallucination evaluation in their experiments, other studies primarily focus on metrics such as precision, recall, F1 score, triple matching, or graph matching. In our approach to evaluating experiments, we consider also hallucination and omission through a linguistic lens.

Upon examining Table 1, we observe the superior performance of the FT method compared to ZSP and FSP for the T2KG construction task. Of particular interest is the finding that, with the exception of Llama2-7b, applying ZSP to the fine-tuned Llama2-7b results in worse performance than FSP on the original Llama2-7b. Overall, this table provides a clear visualization of the relative performance of each method, highlighting the strengths and limitations of each approach for T2KG construction.

Furthermore, it is evident that better results are achieved by providing more examples (more shots) to the same model, whether original or fine-tuned. The results underscore the positive correlation between the quantity of examples and the model's performance. Comparing the fine-tuned Mistral and fine-tuned Starling, they exhibit similar performance when given 7 shots, surpassing the two Llama2 models by a significant margin. The standout performer with ZSP on the fine-tuned LLM is Mistral, showcasing a considerable lead over other LLMs, including Starling. To corroborate these findings, future versions of our study plan to assess our fine-tuned models using an alternative dataset with diverse domains.

As depicted in Figure 2, Hall. represents Hallucinations, while Omis. denotes Omissions.

Taking into account our strategy of introducing an empty graph when LLMs fail to produce triples, we note that even with LLama2-13b with ZSP exhibiting the least favorable results across all metrics, it displays minimal hallucinations. Nonetheless, it's crucial to recognize that the model with the fewest hallucinations may not necessarily be the most suitable choice. To

[2]Hugging Face: https://huggingface.co/

Table 1
Comparison of performance metrics and models

Model | Metric	G-F1	T-F1	G-BS	GED	F1-Bleu	F1-Rouge	Hall.	Omis.
PiVE	14.00	18.57	89.82	11.22	-	-	-	-
Mistral-0	2.30	0.00	77.87	15.93	54.97	55.15	20.63	31.48
Mistral-7	18.72	28.44	87.54	10.13	55.09	63.94	17.88	21.14
Mistral-FT-0	31.93	44.08	86.89	8.25	63.88	69.08	13.55	18.27
Mistral-FT-7	34.68	**49.11**	**91.99**	6.69	**71.78**	**77.43**	15.01	**14.45**
Starling-0	5.23	7.83	86.29	13.35	34.64	14.61	17.48	33.24
Starling-7	21.30	33.77	90.41	8.96	60.47	69.34	17.31	14.61
Starling-FT-0	21.47	28.29	72.86	11.87	44.07	47.69	10.17	42.78
Starling-FT-7	**35.69**	48.49	91.95	**6.60**	71.51	76.67	11.35	18.27
Llama2-7b-0	0.00	0.46	54.20	18.29	20.23	17.98	4.83	81.53
Llama2-7b-7	11.80	20.88	82.78	12.66	45.48	54.29	20.74	30.02
Llama2-7b-FT-0	3.82	15.41	59.19	15.78	16.82	17.95	6.07	79.20
Llama2-7b-FT-7	18.77	32.63	87.19	10.16	58.48	66.35	25.24	18.66
Llama2-13b-0	0.00	0.79	57.42	17.79	20.50	18.23	**4.78**	81.23
Llama2-13b-7	13.49	23.99	84.89	11.59	50.18	58.71	26.36	19.06
Llama2-13b-FT-0	20.52	32.18	75.88	11.38	46.53	50.78	11.64	39.63
Llama2-13b-FT-7	23.55	37.29	88.77	8.94	63.26	70.12	23.55	16.19

overcome this limitation in our evaluation metric, we aim to improve it by considering the prevalence of empty graphs in the generated results before assessing them against ground truth graphs.

The G-BS consistently remains high, indicating that LLMs frequently generate text with words (entities or relations) very similar to those in the ground truth graphs. Among the models, the finetuned Starling with 7 shots achieves the highest G-F1, which focuses on the entirety of the graph and evaluates how many graphs are exactly produced the same, suggesting that it accurately generates approximately 36% of graphs identical to the ground truth. For various metrics, the finetuned Mistral with 7 shots performs exceptionally well, particularly in T-F1, where F1 scores are computed for all test samples and averaged for the final Triple Match F1 score. Additionally, it excels in metrics such as "Omis.," F1-Bleu, and F1-Rouge. F1-Bleu and F1-Rouge represent n-gram-based metrics encompassing precision (Bleu), recall (Rouge), and F-score (Bleu and Rouge). These metric could potentially yield even better results if synonyms of entities or relations are considered as exact matches.

The authors in [6] conduct evaluations using WebNLG+2020. Consequently, we adopt their approach (PiVE) as a baseline for comparison with our experiments. Upon analyzing the results, it becomes evident that nearly all fine-tuned LLMs outperform PiVE, which is applied on both ChatGPT and GPT-4 as mentioned before.

In Table 2, we present the evaluation results of original LLMs with 7 shots and fine-tuned

Table 2
Results on KELM-sub

Model | Metric	G-F1	T-F1	G-BS	GED	F1-Bleu	F1-Rouge	Hall.	Omis.
PiVE	**23.11**	7.50	**87.70**	**11.35**	-	-	-	-
Mistral-7	5.61	10.89	71.29	14.28	56.56	61.11	2.33	77.33
Mistral-FT-0	2.28	8.02	69.29	14.92	24.24	35.70	2.06	77.22
Mistral-FT-7	2.83	8.73	68.55	14.54	26.35	38.76	1.78	78.17
Starling-7	5.61	**13.82**	83.16	12.85	**65.79**	**71.20**	5.33	**59.44**
Starling-FT-0	2.00	5.76	64.87	16.51	17.64	24.29	0.72	79.39
Starling-FT-7	3.11	9.82	67.79	14.53	27.37	39.49	1.22	78.67
Llama2-7b-7	5.06	6.20	67.49	15.55	52.18	56.71	2.28	76.83
Llama2-7b-FT-0	0.22	1.71	58.85	18.84	6.54	7.81	**0.56**	80.28
Llama2-7b-FT-7	5.28	8.33	67.29	15.09	26.86	38.75	3.67	75.33
Llama2-13b-7	5.17	7.82	71.66	15.12	55.39	60.06	3.44	75.56
Llama2-13b-FT-0	1.72	7.73	63.37	15.80	20.59	29.53	1.56	79.44
Llama2-13b-FT-7	4.50	8.63	67.44	14.81	26.33	38.09	2.06	77.22

LLMs with zero-shot and 7 shots on the KELM-sub dataset prepared by [6], building upon [44]. It's crucial to note that the experiments utilized the same prompts as previously described. The 7-shot experiments sourced examples from the WebNLG+2020 training dataset. These new experiments aim to assess the generalization ability of original LLMs with 7 shots and fine-tuned LLMs with zero-shot and 7 shots across diverse domains in the T2KG construction task.

The results in Table 2 indicate that our fine-tuned LLMs perform less effectively than the original LLMs with 7 shots. Furthermore, all LLMs' results on KELM-sub are inferior to those on WebNLG+2020. This disparity can be attributed to the presence of different relation types, where some types are expressed differently in Kelm, utilizing synonyms not considered in the current metrics. Addressing this, our forthcoming versions aim to refine metrics to accommodate synonyms in entities and relations.

We also observe that the evaluation of PiVE on Sub-Kelm yields better results, leveraging examples from the Sub-Kelm training dataset in their few-shot experiments, providing LLMs with insights into certain relation types.

One of the future experimentations will be to use examples from KELM-sub for few-shot prompts to investigate whether the generalization issue stems from WebNLG domains, relation types, or prompts that need improvement to disregard the relation types provided by the examples.

5. Conclusion and perspectives

This study delves into the Text-to-Knowledge Graph (T2KG) construction task, exploring the efficacy of three distinct approaches: Zero-Shot Prompting (ZSP), Few-Shot Prompting (FSP), and Fine-Tuning (FT) of Large Language Models (LLMs). Our comprehensive experimentation, employing models such as Llama2, Mistral, and Starling, sheds light on the strengths and limitations of each approach. The results demonstrate the remarkable performance of the FT method, particularly when compared to ZSP and FSP across various models. Notably, the fine-tuned Llama2-7b with ZSP gaved worst results than FSP with the original Llama2. Additionally, the positive correlation between the quantity of examples and model performance underscores the significance of dataset size in training. An essential part of our study involves the evaluation metrics employed to assess the generated graphs. Particularly, we introduced nuanced considerations for refining these metrics to measuring hallucination and omission in the generated graphs, offering valuable insights into the fidelity of the constructed knowledge graphs.

Looking forward, there are promising perspectives for further enhancement. One is to involve refining evaluation metrics to accommodate synonyms of entities or relations in generated graphs, employing advanced methods or tools for synonym detection. Furthermore, leveraging LLMs for data augmentation in the T2KG construction task shows promise. Notably, during experimentation, LLMs, particularly Starling, exhibited the ability to provide continuity in generated results for T2KG, proposing texts alongside corresponding KGs (triples).

Acknowledgments

The authors thank the French company DAVI (Davi The Humanizers, Puteaux, France) for their support, and the French government for the plan France Relance funding.

References

[1] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, et al., Knowledge graphs, ACM Computing Surveys (Csur) 54 (2021) 1–37.

[2] N. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, J. Taylor, Industry-scale knowledge graphs: Lessons and challenges: Five diverse technology companies show how it's done, Queue 17 (2019) 48–75.

[3] N. Mihindukulasooriya, S. Tiwari, C. F. Enguix, K. Lata, Text2kgbench: A benchmark for ontology-driven knowledge graph generation from text, in: International Semantic Web Conference, Springer, 2023, pp. 247–265.

[4] V. Ershov, A case study for compliance as code with graphs and language models: Public release of the regulatory knowledge graph, arXiv preprint arXiv:2302.01842 (2023).

[5] J. H. Caufield, H. Hegde, V. Emonet, N. L. Harris, M. P. Joachimiak, N. Matentzoglu, H. Kim, S. A. Moxon, J. T. Reese, M. A. Haendel, et al., Structured prompt interrogation and

recursive extraction of semantics (spires): A method for populating knowledge bases using zero-shot learning, arXiv preprint arXiv:2304.02711 (2023).
[6] J. Han, N. Collier, W. Buntine, E. Shareghi, Pive: Prompting with iterative verification improving graph-based generative capability of llms, arXiv preprint arXiv:2305.12392 (2023).
[7] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, L. Zettlemoyer, Rethinking the role of demonstrations: What makes in-context learning work?, arXiv preprint arXiv:2202.12837 (2022).
[8] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al., Training language models to follow instructions with human feedback, Advances in neural information processing systems 35 (2022) 27730–27744.
[9] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, P. F. Christiano, Learning to summarize with human feedback, Advances in Neural Information Processing Systems 33 (2020) 3008–3021.
[10] R. OpenAI, Gpt-4 technical report. arxiv 2303.08774, View in Article 2 (2023).
[11] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.
[12] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., Llama: Open and efficient foundation language models, arXiv preprint arXiv:2302.13971 (2023).
[13] B. Workshop, T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, et al., Bloom: A 176b-parameter open-access multilingual language model, arXiv preprint arXiv:2211.05100 (2022).
[14] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al., Palm: Scaling language modeling with pathways, Journal of Machine Learning Research 24 (2023) 1–113.
[15] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al., Mistral 7b, arXiv preprint arXiv:2310.06825 (2023).
[16] B. Zhu, E. Frick, T. Wu, H. Zhu, J. Jiao, Starling-7b: Improving llm helpfulness & harmlessness with rlaif, 2023.
[17] L. Tunstall, E. Beeching, N. Lambert, N. Rajani, K. Rasul, Y. Belkada, S. Huang, L. von Werra, C. Fourrier, N. Habib, et al., Zephyr: Direct distillation of lm alignment, arXiv preprint arXiv:2310.16944 (2023).
[18] S. Carta, A. Giuliani, L. Piano, A. S. Podda, L. Pompianu, S. G. Tiddia, Iterative zero-shot llm prompting for knowledge graph construction, arXiv preprint arXiv:2307.01128 (2023).
[19] Y. Zhu, X. Wang, J. Chen, S. Qiao, Y. Ou, Y. Yao, S. Deng, H. Chen, N. Zhang, Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities, arXiv preprint arXiv:2305.13168 (2023).
[20] B. Li, G. Fang, Y. Yang, Q. Wang, W. Ye, W. Zhao, S. Zhang, Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness, arXiv preprint arXiv:2304.11633 (2023).
[21] X. Wei, X. Cui, N. Cheng, X. Wang, X. Zhang, S. Huang, P. Xie, J. Xu, Y. Chen, M. Zhang,

et al., Zero-shot information extraction via chatting with chatgpt, arXiv preprint arXiv:2302.10205 (2023).
[22] L. Jarnac, M. Couceiro, P. Monnin, Relevant entity selection: Knowledge graph bootstrapping via zero-shot analogical pruning, in: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, 2023, pp. 934–944.
[23] Z. Bi, J. Chen, Y. Jiang, F. Xiong, W. Guo, H. Chen, N. Zhang, Codekgc: Code language model for generative knowledge graph construction, ACM Transactions on Asian and Low-Resource Language Information Processing 23 (2024) 1–16.
[24] L. Yao, J. Peng, C. Mao, Y. Luo, Exploring large language models for knowledge graph completion, arXiv preprint arXiv:2308.13916 (2023).
[25] H. Khorashadizadeh, N. Mihindukulasooriya, S. Tiwari, J. Groppe, S. Groppe, Exploring in-context learning capabilities of foundation models for generating knowledge graphs from text, arXiv preprint arXiv:2305.08804 (2023).
[26] S. Deng, C. Wang, Z. Li, N. Zhang, Z. Dai, H. Chen, F. Xiong, M. Yan, Q. Chen, M. Chen, et al., Construction and applications of billion-scale pre-trained multimodal business knowledge graph, in: 2023 IEEE 39th International Conference on Data Engineering (ICDE), IEEE, 2023, pp. 2988–3002.
[27] M. Trajanoska, R. Stojanov, D. Trajanov, Enhancing knowledge graph construction using large language models, arXiv preprint arXiv:2305.04676 (2023).
[28] J. Chen, L. Ma, X. Li, N. Thakurdesai, J. Xu, J. H. Cho, K. Nag, E. Korpeoglu, S. Kumar, K. Achan, Knowledge graph completion models are few-shot learners: An empirical study of relation labeling in e-commerce with llms, arXiv preprint arXiv:2305.09858 (2023).
[29] A. Harnoune, M. Rhanoui, M. Mikram, S. Yousfi, Z. Elkaimbillah, B. El Asri, Bert based clinical knowledge extraction for biomedical knowledge graph construction and analysis, Computer Methods and Programs in Biomedicine Update 1 (2021) 100042.
[30] L. Yang, H. Chen, Z. Li, X. Ding, X. Wu, Chatgpt is not enough: Enhancing large language models with knowledge graphs for fact-aware language modeling, arXiv preprint arXiv:2306.11489 (2023).
[31] T. C. Ferreira, C. van der Lee, E. Van Miltenburg, E. Krahmer, Neural data-to-text generation: A comparison between pipeline and end-to-end architectures, arXiv preprint arXiv:1908.09022 (2019).
[32] S. Saha, P. Yadav, L. Bauer, M. Bansal, Explagraphs: An explanation graph generation task for structured commonsense reasoning, arXiv preprint arXiv:2104.07644 (2021).
[33] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, Y. Artzi, Bertscore: Evaluating text generation with bert, arXiv preprint arXiv:1904.09675 (2019).
[34] Z. Abu-Aisheh, R. Raveaux, J.-Y. Ramel, P. Martineau, An exact graph edit distance algorithm for solving pattern recognition problems, in: 4th International Conference on Pattern Recognition Applications and Methods 2015, 2015.
[35] K. Papineni, S. Roukos, T. Ward, W.-J. Zhu, Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th annual meeting of the Association for Computational Linguistics, 2002, pp. 311–318.
[36] C.-Y. Lin, Rouge: A package for automatic evaluation of summaries, in: Text summarization branches out, 2004, pp. 74–81.
[37] C. Gardent, A. Shimorina, S. Narayan, L. Perez-Beltrachini, The webnlg challenge: Gener-

ating text from rdf data, in: Proceedings of the 10th international conference on natural language generation, 2017, pp. 124–133.
[38] A. Hagberg, P. Swart, D. S Chult, Exploring network structure, dynamics, and function using NetworkX, Technical Report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.
[39] T. Dettmers, A. Pagnoni, A. Holtzman, L. Zettlemoyer, Qlora: Efficient finetuning of quantized llms, Advances in Neural Information Processing Systems 36 (2024).
[40] X. Zhang, S. Liu, R. Zhang, C. Liu, D. Huang, S. Zhou, J. Guo, Y. Kang, Q. Guo, Z. Du, et al., Adaptive precision training: Quantify back propagation in neural networks with fixed-point numbers, arXiv preprint arXiv:1911.00361 (2019).
[41] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: Low-rank adaptation of large language models, arXiv preprint arXiv:2106.09685 (2021).
[42] Y. Belkada, T. Dettmers, A. Pagnoni, S. Gugger, S. Mangrulkar, Making llms even more accessible with bitsandbytes, 4-bit quantization and qlora (2023).
[43] S. Mangrulkar, S. Gugger, L. Debut, Y. Belkada, S. Paul, B. Bossan, Peft: State-of-the-art parameter-efficient fine-tuning methods, Younes Belkada and Sayak Paul," PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods (2022).
[44] O. Agarwal, H. Ge, S. Shakeri, R. Al-Rfou, Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training, arXiv preprint arXiv:2010.12688 (2020).