Prompt Me One More Time: A Two-Step Knowledge Extraction Pipeline with Ontology-Based Verification

Alla Chepurova[1] **Yuri Kuratov**[2,1] **Aydar Bulatov**[1] **Mikhail Burtsev**[3]

[1]Neural Networks and Deep Learning Lab, MIPT, Dolgoprudny, Russia
[2]AIRI, Moscow, Russia
[3]London Institute for Mathematical Sciences, London, UK
chepurova@deeppavlov.ai, {bulatov.as,yurii.kuratov}@phystech.edu, mb@lims.ac.uk

Abstract

This study explores a method for extending real-world knowledge graphs (specifically, Wikidata) by extracting triplets from texts with the aid of Large Language Models (LLMs). We propose a two-step pipeline that includes the initial extraction of entity candidates, followed by their refinement and linkage to the canonical entities and relations of the knowledge graph. Finally, we utilize Wikidata relation constraints to select only verified triplets. We compare our approach to a model that was fine-tuned on a machine-generated dataset and demonstrate that it performs better on natural data. Our results suggest that LLM-based triplet extraction from texts, with subsequent verification, is a viable method for real-world applications.

1 Introduction

Today, a vast amount of knowledge exists in unstructured textual formats such as books, articles, news reports, blog posts, and social media. Knowledge graphs (KG), which organize data in a structured form, are crucial for making world knowledge accessible across various applications. One of the largest open real-world knowledge graphs, WikiData (Vrandečić, 2012), contains over 1.54 billion items and is maintained collaboratively by volunteers. Keeping such a resource up-to-date requires significant manual curation. Populating knowledge graphs with information extracted from texts presents a promising solution to this challenge, aiming to automate the process, keep these databases relevant and updated, and help the community support them.

Knowledge graphs (KGs) are directed multi-relational graphs that use entities as nodes and their relationships as edges. To represent KGs, a set of triplets referred to as (head entity, relation, tail entity) or (h, r, t) is used. KGs provide a structured representation of facts regarding both real-world objects and abstract concepts.

Knowledge Extraction, or Triplet Extraction is a crucial task towards automatically constructing large-scale KGs. Such methods are used to identify entity pairs and their relationships in unstructured texts. Three independent steps are usually involved in the KG construction: 1) entity discovery, 2) coreference resolution, and 3) relation extraction. Unfortunately, in this pipeline errors in entity discovery propagate to the subsequent stages limiting overall performance. To address this issue, approaches have recently been developed to jointly extract entities and relations from texts (Melnyk et al., 2021). Such methods tackle both tasks simultaneously as a sequence-to-sequence learning problem in an end-to-end manner using generative language models (LMs). End-to-end generative triple extraction eliminates the issue of error propagation and improves efficiency without requiring additional annotation. However, training a separate LM for a specific KG has several limitations, including 1) requiring labeled corpora of sufficient size for LM training; 2) the need for a re-training model as the KG may actively evolve or LM re-training for a different KG, e.g. KG with distinct ontology, entity, and relation set or a KG from other domain. The last limitation is mediated by the fact that during training LM was provided with only the entities and relations that were present in the previous version of KG or an entirely different KG with a predefined ontology and triplet set.

Large Language Models (LLMs), such as ChatGPT and GPT4, with billions of parameters, are empowering natural language processing with their universal language understanding and generation capabilities, thus creating possibilities for end-to-end KG construction. Although the most advanced methods for generative information extraction depend on fine-tuning sequence-to-sequence models, recent studies (Li et al., 2023) propose that triplet extraction may be possible with LLM by in-context learning (ICL) (Brown et al., 2020) and

Proceedings of TextGraphs-17: Graph-based Methods for Natural Language Processing, pages 61–77

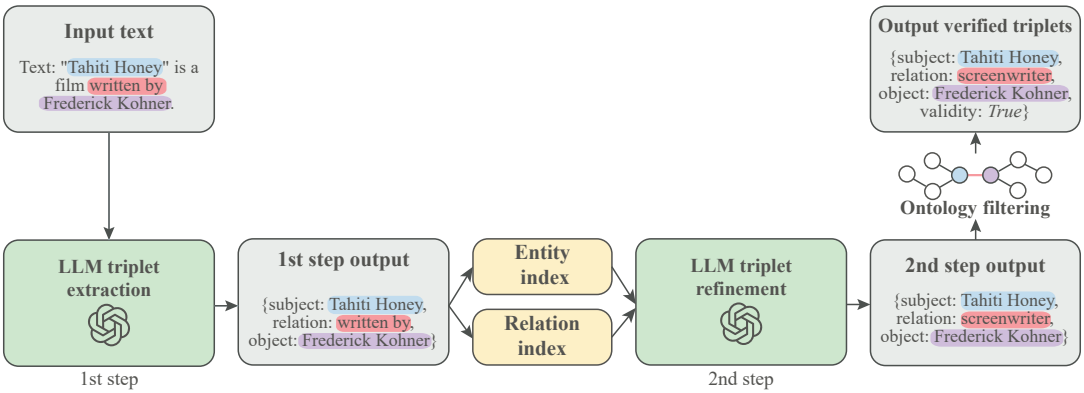

Figure 1: **Proposed pipeline for KG extension.** First, we perform a prompt-based triplet extraction from input text with LLM. The output from this step is a list of triplets in JSON format with possibly inaccurate entity and relation names. Then, we retrieve canonical names of entities and relations for extracted triplets and use them in an LLM prompt to refine triplets. Finally, the refined output is verified with KG ontology to ensure consistency.

instruction following (Ouyang et al., 2022). Using these methods can be advantageous because it eliminates the need for training or fine-tuning models. Thus, triplet extraction can be accomplished by LLMs with the provision of carefully crafted prompts (instructions and in-context examples) and mechanisms for linking names of generated entities and relations to the canonical names from KG.

In this study, we explore the ability of LLMs for tasks of knowledge extraction and KG extension for Wikidata. Despite previous works documenting the poor ability of LLMs to extract structured information from texts (Josifoski et al., 2023), we propose a novel two-step pipeline, which includes 1) LLM to extract entity candidates; 2) LLM to refine triplets by linking exact names of entities and relations based on similar entities and relations from KG; 3) ontology-based triplets verification to enhance the quality of LLM output. Furthermore, we evaluate our pipeline and model that was fine-tuned on the SynthIE dataset to assess their efficacy on real-world data.

2 Methods

To extract triplets from texts, we propose a multi-step pipeline (Figure 1). The first step (Section 2.2) is a candidate triplets extraction. Triplet candidates may include entities and relation names that do not directly match the formats used in the WikiData KG. Therefore, the second step (Section 2.3) refines these candidates based on similar entities and relations that are present in the KG. Finally, we verify the refined triplets and filter out those that are not consistent with the KG ontology (Section 2.4). This involves checking relation constraints and ensuring compatibility of entity types. We use the OpenAI `gpt-3.5-turbo` model to implement the first two steps of the pipeline by providing in-context examples and instructions.

2.1 Datasets

Gathering datasets for the triplet extraction task is both time-consuming and costly as annotators must be familiar with all the entities and relations from a KG to reason about every potential fact stated in the text. In the case of large real-world KGs such as Wikidata, it represents a substantial challenge. This leads to the lack of quality datasets for the KG construction task, while only sparse or noisy datasets are available. For instance, texts in the largest accessible dataset, REBEL (Huguet Cabot and Navigli, 2021), frequently lack extractable information about entities in the text, instead substituting pronouns for factual information about entities. Moreover, target triplets in REBEL also do not contain all the facts provided in the input or are partially inaccurate (Josifoski et al., 2023). Other popular datasets for this task have similar problems (Josifoski et al., 2021).

Evaluation on distantly supervised datasets like REBEL would therefore give an extremely erroneous assessment of the models' performance. Therefore, we used a higher quality SynthIE (Josifoski et al., 2023) dataset. Despite the fact that the dataset was synthetically generated, authors claimed LLMs' inability to solve the information extraction task. Therefore, the task was considered asymmetrical since LLMs can produce text from KG triplets but not KG from text.

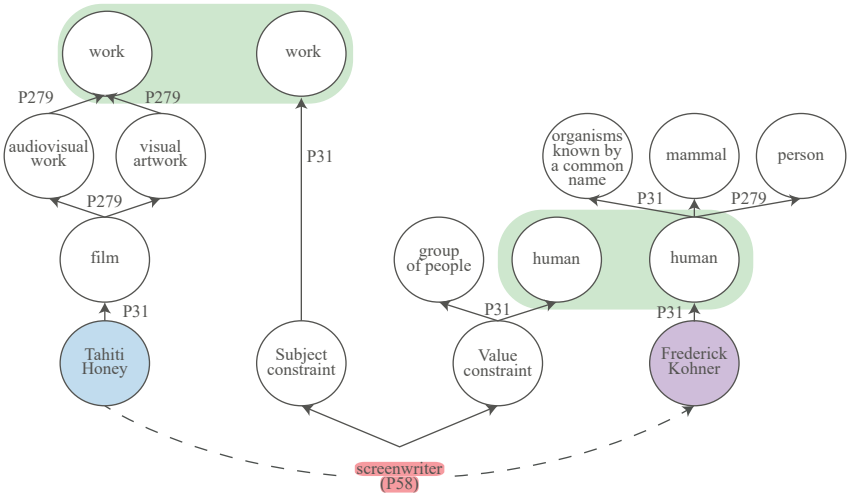

Figure 2: **Ontology-based verification.** Triplets generated in the proposed pipeline undergo ontology verification aimed at ensuring consistency between the output and KG ontology. Triplet claimed to be valid in case its subject's and object's hierarchy types intersect with the types from relation constraints. The ontology hierarchy is simplified for demonstration purposes. P31 and P279 stand for *"instance of"* and *"subclass of"* relations correspondingly.

2.2 Extraction of triplet candidates

In the first step, we extract triplet candidates from the provided text. We benchmarked OpenAI `gpt-3.5-turbo`[1] on triplet extraction with in-context examples. In the prompt, we described the essence of the task and used three examples from the train split from the Wiki-cIE Code dataset (Josifoski et al., 2023) for demonstration. We provide details on prompts construction in Appendix A. In this step, for example, one of the triplets extracted by LLM from the text *"Tahiti Honey" is a film written by Frederick Kohner* would be: *("Tahiti Honey", "written by", "Frederick Kohner")*.

However, the subject, object, and relation in the extracted triplet are not necessarily normalized, i.e. they could be inconsistent with the name conventions used in the original KG.

2.3 Refinement of triplet candidates

The primary limitation of LLMs applied to KG construction is that, despite their ability to extract the essential information from a text, they are still unable to link extracted names with the canonical ones of Wikidata entities or relations. To address this limitation, we used a two-step LLM prompting strategy. After extracting information from the text on the first step, the FAISS (Johnson et al., 2019) index was used to retrieve canonical names from the Wikidata KG, ranked by cosine similarity to the entities and relations from identified triplets. The index was built using pre-trained Contriever embeddings (Izacard et al., 2021) that demonstrated robustness and strong performance across various retrieval scenarios. Based on the top-5 retrieved exact names of entities and relations similar to the ones extracted in the first step, we define the task of choosing the names that fit the context of a text and triplet itself. The full structure of the second step prompt is described in Appendix A. For example, for the above-mentioned triplet, there would be retrieved 5 canonical names ranked by similarity to the extracted one for both subject and object entities and relation:

"written by": ["lyrics by", "adapted by", "produced by", "screenwriter", "author"],
"Tahiti Honey": ["Tahiti Honey", "Honey", "Honey Chile", "Celtic Honey", "Tahitipresse"],
"Frederick Kohner": ["Frederick Kohner", "Paul Kohner", "Adolf Kohner", "Susan Kohner", "Henry Rohner"].

As a result, after choosing the corresponding names that fit the context of the text, the refined triplet is *("Tahiti Honey", "screenwriter", "Frederick Kohner")*.

2.4 Ontology-based verification

To increase the reliability of the pipeline outputs to possible LLM hallucinations, we proposed an automatic verification of generated triplets based on the KG ontology. An ontology is a semantical model for knowledge representation in a specific domain, which specifies types of entities represented in this

[1] `gpt-3.5-turbo-0125`: https://platform.openai.com/docs/models/gpt-3-5

domain as well as constraints regulating how the entities can interact through relations. To ensure the consistency between generated triplets and the WikiData KG ontology model, we used information about types of subjects and objects from generated triplets and relation constraints that specify which types of entities can be connected through the extracted relations.

After both stages of prompting, output triplets underwent an automated check on semantics and property constraints available in Wikidata KG. For this purpose, we used *subject* (Q21503250) and *value* (Q21510865) constraints of the extracted relation. These constraints are attributed to a specific relation and declare which type of head and tail entity correspondingly should be used in a triplet with this relation. Further, classes of both subject and value entities from generated triplet were extracted from Wikidata Query Service (WDQS) by querying *"instance of"* (P31) and *"subclass of"* (P279) properties of subject and object up to the root of their subclass hierarchy. The full SPARQL queries used to retrieve WDQS are described in Appendix B. A triplet is considered valid if the hierarchical types of both the subject and object align with the types specified in the relation constraints. In this way, the validity of the logical structure of extracted information was ensured. An example of the ontology verification process is schematically described in Figure 2.

3 Results and discussion

3.1 Proposed pipeline improves LLM KG extension performance

We suggest that triplet extraction can be accomplished by LLMs with the provision of carefully designed prompts and refining mechanisms for converting imprecise names of generated outputs to the canonical ones from KG. Table 1 shows performance metrics of the full and ablated method.

It is worth noting that the most crucial part of the pipeline is providing the LLM with representative examples of triplets. Using two-step prompting with example demonstration and triplet verification results in a five times better F1 score compared to the same pipeline without example demonstration.

Enhancing the pipeline with the triplet refinement step significantly improves the recall score compared to the single-prompt approach. In turn, the verification step is essential to retain the precision score after triplet refinement comparable to one obtained solely after the first step. In a single-prompt setup, verification does not provide significant improvement. The explanation for this is that after the first step, all triplets whose names KG lacks are automatically filtered out, thus leaving only those that have been referred to most accurately in the text, so LLM can identify them precisely. In the second step, however, LLM may lack knowledge of the specific KG ontology while composing new triplets with exact names of entities and relations that were originally inaccurately specified in the text. Providing ontologies in the second prompt would make it unreasonably large, so we used filtering in the post-processing step to retain a higher precision score, while not reducing the recall value.

3.2 Synthetic data is not a cure-all

Table 2 demonstrates the score of the entire proposed pipeline compared to SynthIE T5-large, the best-reported model trained on SynthIE dataset (Josifoski et al., 2023). Although the improved two-step pipeline with verification and in-context examples demonstrates the potential to improve the quality of KG construction with LLM, its performance on synthetic data compared to the pre-trained model appears rather modest.

In turn, by manually reviewing the synthetic dataset, we observed frequent inconsistencies between the texts and the target triplets, examples of which are provided in Appendix C. To demonstrate the shortcomings of the synthetically generated dataset, we chose a strategy different from utilized in (Josifoski et al., 2023) for REBEL. Instead of assessing the whole dataset itself, we selected 100 random samples from the WikicIE-test-small. Employing the reference entities used for the generation of each text, we manually added similar texts in natural language about these entities from Wikipedia paragraphs. In this way, we obtained a set of texts in natural language referring to the KG entities from the original dataset. Both LM trained on the synthetic data and our LLM-based pipeline generated triplets based on these texts. Then, LM's and LLM's outputs were subjected to human evaluation, described in detail in Appendix D. Results of human evaluation are presented in Table 2. LLM-based pipeline outperforms SynthIE T5-large by a wide margin in terms of human-evaluated precision. The pre-trained model tends to generate triplets that do not directly fit the context of the text, using learned triplets instead of extracting rel-

Table 1: The LLM-based two-stage strategy for triplet refinement and ontology verification increases performance in KG construction tasks. We report mean and std of three runs with different in-context examples.

Method	Metrics (SynthIE-text-small)			Conclusion
	F1	Precision	Recall	
Full pipeline	**0.55**↑±0.01	0.59↑±0.03	**0.52** ±0.005	
Full pipeline ǀ no verification	0.53 ±0.01	0.55±0.03	**0.52**±0.005	Verification helps
Full pipeline ǀ no refinement	0.51±0.01	0.60↑±0.03	0.44 ±0.006	Refinement helps
Full pipeline ǀ no refinement ǀ no verification	0.50 ±0.01	0.58±0.03	0.44 ±0.006	Verification helps
Full pipeline ǀ no in-context examples	0.16	**0.65**	0.09	In-context examples help

Table 2: Our proposed two-step approach outperforms the SynthIE model on a natural language set with human evaluation. While it does not outperform the SynthIE model tuned directly on the SynthIE-text-small dataset and learned its specifics, low IoU metric indicate that the two methods produce very different predictions. (*) results were obtained by the model fine-tuned on the dataset.

Model	SynthIE-text-small			Natural Language corpora		
	F1	Precision	Recall	Precision	# Correct	IoU
Prompt me one more time (Ours)	0.55 ±0.01	0.59 ±0.03	0.52 ±0.005	0.74	187	0.08
SynthIE T5-large (Josifoski et al., 2023)	0.88*	0.87*	0.88*	0.55	201	

evant ones from the context. The reason for this is the quality of the dataset used for training and for in-context learning in our pipelines, due to the inconsistencies between triplets in the annotation and input text.

It is also worth noting that the correct triplets predicted by the two models exhibit notable differences from each other. The intersection over union (IoU) metric shows that only 8 percent of correct triplets are predicted by both models. This suggests that each model has a different area of specialization, excelling in different aspects of information extraction. Consequently, their predictions may be combined, yielding further improvements in the pipeline performance.

4 Conclusions and Future work

Our study identified that LLMs coupled with a two-stage strategy for triplet refinement and ontology verification are competitive in extracting triplets from texts. Furthermore, the proposed LLM-based pipeline shows better performance on non-synthetic data, compared to the model fine-tuned on the SynthIE dataset, making our approach a promising way to address the extraction of triplets from real-world data.

Exploration of possible improvement mechanisms for the triplet refinement step could be a focus of further studies to obtain quality comparable with methods fine-tuned for this task. Current entity and relation linking uses cosine similarity on pre-trained embeddings applied to canonical names from Wikidata. However, enhancing retrieval augmented part is crucial to increase the recall of the whole system. Hence further research may focus on fine-tuning embeddings for this task and the use of additional information from Wikidata, e.g. entities from node neighborhood (Kochsiek et al., 2023; Chepurova et al., 2023), entity or relation textual description.

We identified drawbacks in the synthetic dataset, which was established as one with superior quality in prior work (Josifoski et al., 2023). Considering the demonstrated potential of LLM applied to a triplet extraction task, the creation of smaller, yet higher quality benchmarks for LLM based on texts expressed in natural language could resolve challenges of dataset collection in terms of time and cost and eliminate a bias introduced by synthetic generation.

Limitations

Fine-tuning Overhead. Despite the significant performance improvement achieved by our two-stage pipeline with in-context examples, the models directly fine-tuned for specific graphs and datasets may still outperform our approach. This means that in cases where the text and information domain are sufficiently narrow and computational resources are available, specialized models may be preferred.

Domain Specifics. Furthermore, the effectiveness of the proposed pipeline may vary depending on the domain of source text and knowledge graph. Particularly challenging is dealing with texts that

are significantly different from the pre-training domain. The domain adaptation may require manual intervention such as augmentation of in-context examples with domain-specific terms.

Prompt Sensitivity. Our experiments have revealed the considerable influence that prompt quality exerts on the overall performance of the pipeline. This underscores the significance of prompt selection, as it directly affects the pipeline performance, particularly when extending its applicability to new domains.

Acknowledgements

This work was supported by a grant for research centers, provided by the Analytical Center for the Government of the Russian Federation in accordance with the subsidy agreement (agreement identifier 000000D730324P540002) and the agreement with the Moscow Institute of Physics and Technology dated November 1, 2021 No. 70-2021-00138.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Alla Chepurova, Aydar Bulatov, Yuri Kuratov, and Mikhail Burtsev. 2023. Better together: Enhancing generative knowledge graph completion with language models and neighborhood information. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5306–5316, Singapore. Association for Computational Linguistics.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. Rebel: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, Online and in the Barceló Bávaro Convention Centre, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Martin Josifoski, Nicola De Cao, Maxime Peyrard, Fabio Petroni, and Robert West. 2021. Genie: Generative information extraction. *arXiv preprint arXiv:2112.08340*.

Martin Josifoski, Marija Sakota, Maxime Peyrard, and Robert West. 2023. Exploiting asymmetry for synthetic training data generation: SynthIE and the case of information extraction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1555–1574, Singapore. Association for Computational Linguistics.

Adrian Kochsiek, Apoorv Saxena, Inderjeet Nair, and Rainer Gemulla. 2023. Friendly neighbors: Contextualized sequence-to-sequence link prediction. In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 131–138, Toronto, Canada. Association for Computational Linguistics.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023. Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *arXiv preprint arXiv:2304.11633*.

Igor Melnyk, Pierre Dognin, and Payel Das. 2021. Grapher: Multi-stage knowledge graph construction using pretrained language models. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Denny Vrandečić. 2012. Wikidata: a new platform for collaborative data collection. In *Proceedings of the 21st International Conference on World Wide Web*, page 1063–1064, New York, NY, USA. Association for Computing Machinery.

A Prompts

We employed a two-step prompting strategy to 1) extract structural information in the form of triplets from a text, and 2) link imprecise names of entities and relations resulting in the first step to the canonical Wikidata names and ids. The output from the first step was parsed as a JSON object to obtain a list of extracted triplets. Then, relations and entities in those triplets were linked to the corresponding top-5 similar exact names from Wikidata using the FAISS index as it was described in 2.3. This resulted in the mapping of extracted triplets and the top-5 exact names were used to construct the second prompt.
For a few-shot prompting with in-context examples, we utilized several crafted examples from a training split of the Wiki-cIE Code dataset. We used different combinations of such examples in three launches, and 3 different sets of examples were taken for each launch of our pipeline.

A.1 System prompts

Triplet retrieval prompt

```
You are an algorithm designed for extracting facts from text in a structured format to build a knowledge graph.
Knowledge graphs consists of a set of triplets. Each triplet contains two entities (subject and object) and one relation
that connects these subject and object.
Entities represent nodes in the knowledge graph, while relation represents a link between these two nodes.
Subjects and objects could be named entities or concepts describing a group of people, events,or abstract objects
from the Wikidata knowledge graph.

You will be provided with the text entitled "Text:". You are expected to output only the list of identified triplets
in a JSON format. Each triplet should have fields "subject", "relation", and "object" for subject, relation, and object
correspondingly.

Here are a few examples of input texts and expected output for each of them:

<example>
...
</example>
```

Triplet refinement prompt

```
In the previous step, there were extracted triplets from the Wikidata knowledge graph.
Each triplet contains two entities (subject and object) and one relation that connects these subject and object.
However, some of the entities and relations extracted in the previous step may have not an exact name from Wikidata.
We linked each subject, relation, and object name with top similar exact names from the Wikidata by semantic similarity.

Your task is to choose appropriate names for entities and relations that correspond to the text's context and
triplet they were taken from.

The input format is following:
    Text: <Text from the previous step for triplets extraction>

    Triplets and corresponding entity and relation mappings:

        <triplet_1>
        {<extracted_subject_1>: <list of 5 exact names similar to subject_1>}
        {<extracted_relation_1>: <list of 5 exact names similar to relation_1>}
        {<extracted_object_1>: <list of 5 exact names similar to object_1>}

        ...

        <triplet_n>
        {<extracted_subject_n>: <list of 5 exact names similar to subject_n>}
        {<extracted_relation_n>: <list of 5 exact names similar to relation_n>}
        {<extracted_object_n>: <list of 5 exact names similar to object_n>}

For each triplet obtained in the previous step, you will be provided with the corresponding 5 exact names of similar
subjects, relations, and objects from the Wikidata knowledge graph.
For each subject, relation, and object from the extracted triplets, you should choose the most appropriate name only from
the corresponding list of 5 exact ones that better match each triplet and the context of the previously demonstrated text.
For each relation and entity in the triplets use chosen names only from the provided list of exact similar names.
You can keep the original name unchanged only in case it is already in the list and fits the context of the text
and extracted triplet better than other options.
In the output provide only the list of updated triplets in JSON format. Each triplet should have fields "subject",
"relation", and "object" for updated subject, relation, and object correspondingly.

Here are some examples of previously extracted triplets with imprecise names, a listsd of linked
exact names, and corresponding corrected outputs:

<example>
...
</example>
```

A.2 In-context examples for the first step prompting

Examples for the first step prompting

```
<example>
Input:
        Text: Józef Skrobiński was a Polish film director and member of the Association of Polish Artists and Designers,
    located in Warsaw, Poland.

Output:
        [
                {
                        "subject": "Józef Skrobiński",
                        "relation": "languages spoken, written or signed",
                        "object": "Polish language"
                },
                {
                        "subject": "Józef Skrobiński",
                        "relation": "occupation",
                        "object": "Film director"
                },
                {
                        "subject": "Józef Skrobiński",
                        "relation": "member of",
                        "object": "Association of Polish Artists and Designers"
                },
                {
                        "subject": "Józef Skrobiński",
                        "relation": "country of citizenship",
                        "object": "Poland"
                },
                {
                        "subject": "Józef Skrobiński",
                        "relation": "instance of",
                        "object": "Human"
                },
                {
                        "subject": "Association of Polish Artists and Designers",
                        "relation": "located in the administrative territorial entity",
                        "object": "Warsaw"
                },
                {
                        "subject": "Association of Polish Artists and Designers",
                        "relation": "country",
                        "object": "Poland"
                }
        ]
</example>

<example>
Input:
        Text: Jorge Merino is a Spanish footballer who plays as a midfielder for Marino de Luanco.

Output:
        [
                {
                        "subject": "Jorge Merino",
                        "relation": "languages spoken, written or signed",
                        "object": "Spanish language"
                },
                {
                        "subject": "Jorge Merino",
                        "relation": "sport",
                        "object": "Association football"
                },
                {
                        "subject": "Jorge Merino",
                        "relation": "occupation",
                        "object": "Football player"
                },
                {
                        "subject": "Jorge Merino",
                        "relation": "position played on team / speciality",
                        "object": "Midfielder"
                },
                {
                        "subject": "Jorge Merino",
                        "relation": "member of sports team",
                        "object": "Marino de Luanco"
                },
                {
                        "subject": "Jorge Merino",
                        "relation": "country of citizenship",
                        "object": "Spain"
                },
                {
                        "subject": "Midfielder",
                        "relation": "subclass of",
                        "object": "Football player"
                },
                {
                        "subject": "Midfielder",
                        "relation": "instance of",
                        "object": "Association football positions"
                }
        ]
</example>
```

Examples for the first step prompting

```
<example>
Input:
        Text: Mario Griguol is an Argentine football player born in Córdoba, Argentina. He speaks Spanish and plays for
    Club Atlético Atlanta.

Output:
        [
                {
                        "subject": "Mario Griguol",
                        "relation": "sport",
                        "object": "Association football"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "occupation",
                        "object": "Football player"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "place of birth",
                        "object": "Córdoba, Argentina"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "country of citizenship",
                        "object": "Argentina"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "languages spoken, written or signed",
                        "object": "Spanish language"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "member of sports team",
                        "object": "Club Atlético Atlanta"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "instance of",
                        "object": "Human"
                },
        ]
</example>

<example>
Input:
        Text: "Tahiti Honey" is an English-language film written by Frederick Kohner.

Output:
        [
                {
                        "subject": "Tahiti Honey",
                        "relation": "original language of film or TV show",
                        "object": "English language"
                },
                {
                        "subject": "Tahiti Honey",
                        "relation": "instance of",
                        "object": "Film"
                },
                {
                        "subject": "Tahiti Honey",
                        "relation": "screenwriter",
                        "object": "Frederick Kohner"
                }
        ]
</example>

<example>
Input:
        Text: Lexington Avenue is a thoroughfare in New York City, parallel to Park Avenue and Third Avenue.

Output:
        [
                {
                        "subject": "Lexington Avenue",
                        "relation": "instance of",
                        "object": "Thoroughfare"
                },
                {
                        "subject": "Lexington Avenue",
                        "relation": "parallel street",
                        "object": "Park Avenue"
                },
                {
                        "subject": "Lexington Avenue",
                        "relation": "parallel street",
                        "object": "Third Avenue"
                },
        ]
</example>
```

Examples for the first step prompting

```
<example>
Input:
        Text: ArXiv is funded by Los Alamos National Laboratory and Nagoya University.

Output:
        [
                {
                        "subject": "ArXiv",
                        "relation": "funder",
                        "object": "Los Alamos National Laboratory"
                },
                {
                        "subject": "ArXiv",
                        "relation": "funder",
                        "object": "Nagoya University"
                },
        ]
</example>
```

A.3 In-context examples for the second step prompting

Examples for the second step prompting

```
<example>
Input:
    Text: Mario Griguol is an Argentine football player born in Córdoba, Argentina. He speaks Spanish and plays for
    Club Atlético Atlanta.

    Triplets and corresponding entity and relation mappings:

        {"subject": "Mario Griguol", "relation": "sport", "object": "football"}
        {"Mario Griguol": ["Mario Griguol", "Carlos Timoteo Griguol", "Roberto Grigis", "M\u00e1rio Grman", "Mario Gromo"]}
        {"sport": ["sport", "sport number", "country for sport", "sports league level",
        "sports season of league or competition"]}
        {"football": ["Association football", "Football association", "The Football Association", "Football",
        "Association football positions"]}

        {"subject": "Mario Griguol", "relation": "occupation", "object": "footballer"}
        {"Mario Griguol": ["Mario Griguol", "Carlos Timoteo Griguol", "Roberto Grigis", "M\u00e1rio Grman", "Mario Gromo"]}
        {"occupation": ["occupation", "field of this occupation", "enclave within", "territory claimed by", "residence"]}
        {"footballer": ["Football player", "Football", "Football on 5", "Football at the Summer Olympics",
        "American football"]}

        {"subject": "Mario Griguol", "relation": "born in", "object": "Córdoba, Argentina"}
        {"Mario Griguol": ["Mario Griguol", "Carlos Timoteo Griguol", "Roberto Grigis", "M\u00e1rio Grman", "Mario Gromo"]}
        {"born in": ["place of birth", "family", "birth name", "family name", "presenter"]}
        {"Córdoba, Argentina": ["C\u00f3rdoba, Argentina", "C\u00f3rdoba F.C.", "C\u00f3rdoba, Spain",
        "C\u00f3rdoba Province, Argentina", "C\u00f3rdoba Department"]}

        {"subject": "Mario Griguol", "relation": "citizenship", "object": "Argentina"}
        {"Mario Griguol": ["Mario Griguol", "Carlos Timoteo Griguol", "Roberto Grigis", "M\u00e1rio Grman", "Mario Gromo"]}
        {"citizenship": ["country of citizenship", "allegiance", "diaspora", "country of origin", "flag"]}
        {"Argentina": ["Argentina", "Norma Argentina", "Argentine Northwest", "Argentina Classic", "Argentine Islands"]}

        {"subject": "Mario Griguol", "relation": "spoken language", "object": "Spanish language"}
        {"Mario Griguol": ["Mario Griguol", "Carlos Timoteo Griguol", "Roberto Grigis", "M\u00e1rio Grman", "Mario Gromo"]}
        {"spoken language": ["official language", "native language", "language used", "languages spoken, written or signed",
        "dialect of"]}
        {"Spanish language": ["Spanish language", "Standard Spanish", "Spanish language in the United States",
        "Mexican Spanish", "UN Spanish Language Day"]}

        {"subject": "Mario Griguol", "relation": "member of team", "object": "Club Atlético Atlanta"}
        {"Mario Griguol": ["Mario Griguol", "Carlos Timoteo Griguol", "Roberto Grigis", "M\u00e1rio Grman", "Mario Gromo"]}
        {"member of team": ["member of sports team", "member of the crew of", "member of military unit", "member of",
        "part of"]}
        {"Club Atlético Atlanta": ["Club Atl\u00e9tico Atlanta", "Club Atl\u00e9tico Palermo",
        "Club Atl\u00e9tico Nueva Chicago", "Atl\u00e9tico Boca del R\u00edo", "Club Atl\u00e9tico Del Plata"]}

        {"subject": "Mario Griguol", "relation": "is", "object": "Human"}
        {"Mario Griguol": ["Mario Griguol", "Carlos Timoteo Griguol", "Roberto Grigis", "M\u00e1rio Grman", "Mario Gromo"]}
        {"is": ["represents", "instance of", "replaces", "contains", "is a list of"]}
        {"Human": ["Human", "Humanos", "Human Is", "Human Entertainment", "Humanin"]}


Output:
        [
                {
                        "subject": "Mario Griguol",
                        "relation": "sport",
                        "object": "Association football"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "occupation",
                        "object": "Football player"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "place of birth",
                        "object": "Córdoba, Argentina"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "country of citizenship",
                        "object": "Argentina"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "languages spoken, written or signed",
                        "object": "Spanish language"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "member of sports team",
                        "object": "Club Atlético Atlanta"
                },
                {
                        "subject": "Mario Griguol",
                        "relation": "instance of",
                        "object": "Human"
                },
        ]
</example>
```

Examples for the second step prompting

```
<example>
Input:
    Text: Józef Skrobiński was a Polish film director and member of the Association of Polish Artists and Designers, located in Warsaw, Poland.

    Triplets and corresponding entity and relation mappings:

        {"subject": "Józef Skrobiński", "relation": "spoken language", "object": "Polish"}
        {"Józef Skrobiński": ["Józef Skrobiński", "Józef Żabiński", "Antoni Czubiński", "Jan Żabiński", "Bezek Dębiński"]}
        {"spoken language": ["official language", "native language", "language used", "languages spoken", "written or signed", "dialect of"]}
        {"Polish": ["Polish language", "Dialects of Polish", "Polish alphabet", "Warsaw dialect", "Polish manual alphabet"]}

        {"subject": "Józef Skrobiński", "relation": "occupation", "object": "film director"}
        {"Józef Skrobiński": ["Józef Skrobiński", "Józef Żabiński", "Antoni Czubiński", "Jan Żabiński", "Bezek Dębiński"]}
        {occupation: ["occupation", "field of this occupation", "enclave within", "territory claimed by", "residence"]}
        {"film director": ["Film director", "Film producer", "Bi Gan (film director)", "Madan (film director)", "Television director"]}

        {"subject": "Józef Skrobiński", "relation": "member", "object": "Association of Polish Artists and Designers"}
        {"Józef Skrobiński": ["Józef Skrobiński", "Józef Żabiński", "Antoni Czubiński", "Jan Żabiński", "Bezek Dębiński"]}
        {"member": ["member of", "member of sports team", "member of military unit", "member count", "member of the crew of"]}
        {"Association of Polish Artists and Designers": ["Association of Polish Artists and Designers", "Association of Polish Architects",
        "Polish Association of Artists – 'The Capitol'", "Polish pavilion", "Polish Social and Cultural Association"]}

        {"subject": "Józef Skrobiński", "relation": "citizenship", "object": "Poland"}
        {"Józef Skrobiński": ["Józef Skrobiński", "Józef Żabiński", "Antoni Czubiński", "Jan Żabiński", "Bezek Dębiński"]}
        {"citizenship": ["country of citizenship", "country of origin", "country of registry", "place of birth", "place of origin"]}
        {"Poland": ["Poland", "Ł", "Polesia", "Poland Together", "Poland Railroad Station"]}

        {"subject": "Józef Skrobiński", "relation": "is", "object": "human"}
        {"Józef Skrobiński": ["Józef Skrobiński", "Józef Żabiński", "Antoni Czubiński", "Jan Żabiński", "Bezek Dębiński"]}
        {"is": ["represents", "instance of", "replaces", "contains", "is a list of"]}
        {"human": ["Human", "Humanos", "Human Is", "Human Entertainment", "Humanin"]}

        {"subject": "Association of Polish Artists and Designers", "relation": "location", "object": "Warsaw"}
        {"Association of Polish Artists and Designers": ["Association of Polish Artists and Designers", "Association of Polish Architects",
        "Polish Association of Artists – 'The Capitol'", "Polish pavilion", "Polish Social and Cultural Association"]}
        {"location": ["location", "work location", "coordinate location", "located in the administrative territorial entity",
        "terminus location"]}
        {"Warsaw": ["Warsaw", "Warsaw Confederation", "Tiger Warsaw", "Warsaw Derby", "Warsaw Arsenal"]}

        {"subject": "Association of Polish Artists and Designers", "relation": "country", "object": "Poland"}
        {"Association of Polish Artists and Designers": ["Association of Polish Artists and Designers", "Association of Polish Architects",
        "Polish Association of Artists – 'The Capitol'", "Polish pavilion", "Polish Social and Cultural Association"]}
        {"country": ["country", "country for sport", "basin country", "country of citizenship", "head of state"]}
        {"Poland": ["Poland", "Ł", "Polesia", "Poland Together", "Poland Railroad Station"]}

Output:
    [
        {
            "subject": "Józef Skrobiński",
            "relation": "languages spoken, written or signed",
            "object": "Polish language"
        },
        {
            "subject": "Józef Skrobiński",
            "relation": "occupation",
            "object": "Film director"
        },
        {
            "subject": "Józef Skrobiński",
            "relation": "member of",
            "object": "Association of Polish Artists and Designers"
        },
        {
            "subject": "Józef Skrobiński",
            "relation": "country of citizenship",
            "object": "Poland"
        },
        {
            "subject": "Józef Skrobiński",
            "relation": "instance of",
            "object": "Human"
        },
        {
            "subject": "Association of Polish Artists and Designers",
            "relation": "located in the administrative territorial entity",
            "object": "Warsaw"
        },
        {
            "subject": "Association of Polish Artists and Designers",
            "relation": "country",
            "object": "Poland"
        },
    ]
</example>
```

Examples for the second step prompting

```
<example>
Input
    Text: Jorge Merino is a Spanish footballer who plays as a midfielder for Marino de Luanco.

    Triplets and corresponding entity and relation mappings:

        {"subject": "Jorge Merino", "relation": "spoken language", "object": "Spanish"}
        {"Jorge Merino": ["Jorge Merino", "Luis Merino", "Juan Merino", "Roberto Merino", "Pedro Merino"]}
        {"spoken language": ["official language", "native language", "language used", "languages spoken", "written or signed", "dialect of"]}
        {"Spanish": ["Spanish language", "Standard Spanish", "Spanish language in the United States", "Mexican Spanish",
        "UN Spanish Language Day"]}

        {"subject": "Jorge Merino", "relation": "sport", "object": "football"}
        {"Jorge Merino": ["Jorge Merino", "Luis Merino", "Juan Merino", "Roberto Merino", "Pedro Merino"]}
        {"sport": ["sport", "sport number", "country for sport", "sports league level", "sports season of league or competition"]}
        {"football": ["Association football", "Football association", "The Football Association", "Football", "Association football positions"]}

        {"subject": "Jorge Merino", "relation": "occupation", "object": "footballer"}
        {"Jorge Merino": ["Jorge Merino", "Luis Merino", "Juan Merino", "Roberto Merino", "Pedro Merino"]}
        {"occupation": ["occupation", "field of this occupation", "enclave within", "territory claimed by", "residence"]}
        {"footballer": ["Football player", "Football", "Football on 5", "Football at the Summer Olympics", "American football"]}

        {"subject": "Jorge Merino", "relation": "position", "object": "midfielder"}
        {"Jorge Merino": ["Jorge Merino", "Luis Merino", "Juan Merino", "Roberto Merino", "Pedro Merino"]}
        {"position": ["position held", "position played on team / speciality", "direction", "academic appointment", "military rank"]}
        {"midfielder": ["Midfielder", "Liga Forward", "Tó (footballer)", "FC Copa", "Defender 2000"]}

        {"subject": "Jorge Merino", "relation": "member of team", "object": "Marino de Luanco"}
        {"Jorge Merino": ["Jorge Merino", "Luis Merino", "Juan Merino", "Roberto Merino", "Pedro Merino"]}
        {"member of team": ["member of sports team", "member of the crew of", "member of military unit", "member of", "part of"]}
        {"Marino de Luanco": ["Marino de Luanco", "Luanco", "Luan Cândido", "Luan Gabriel", "Luan Sérgio"]}

        {"subject": "Jorge Merino", "relation": "citizenship", "object": "Spain"}
        {"Jorge Merino": ["Jorge Merino", "Luis Merino", "Juan Merino", "Roberto Merino", "Pedro Merino"]}
        {"citizenship": ["country of citizenship", "member of military unit", "official residence", "place of birth", "member of the crew of"]}
        {"Spain": ["Spain", "Spain Rodriguez", "Spanish City", "Madrid", "María", "Spain"]}

        {"subject": "midfielder", "relation": "subclass", "object": "footballer"}
        {"midfielder": ["Midfielder", "Liga Forward", "Tó (footballer)", "FC Copa", "Defender 2000"]}
        {"subclass": ["subclass of", "is metaclass for", "has parts of the class", "social classification", "competition class"]}
        {"footballer": ["Football player", "Football", "Football on 5", "Football at the Summer Olympics", "American football"]}

        {"subject": "midfielder", "relation": "instance of", "object": "football position"}
        {"midfielder": ["Midfielder", "Liga Forward", "Tó (footballer)", "FC Copa", "Defender 2000"]}
        {"instance of": ["instance of", "part of", "follower of", "part of the series", "member of"]}
        {"football position": ["Association football positions", "American football positions", "Australian rules football positions",
        "Basketball positions", "Baseball positions"]}

Output:
    [
        {
            "subject": "Jorge Merino",
            "relation": "languages spoken, written or signed",
            "object": "Spanish language"
        },
        {
            "subject": "Jorge Merino",
            "relation": "sport",
            "object": "Association football"
        },
        {
            "subject": "Jorge Merino",
            "relation": "occupation",
            "object": "Football player"
        },
        {
            "subject": "Jorge Merino",
            "relation": "position played on team / speciality",
            "object": "Midfielder"
        },
        {
            "subject": "Jorge Merino",
            "relation": "member of sports team",
            "object": "Marino de Luanco"
        },
        {
            "subject": "Jorge Merino",
            "relation": "country of citizenship",
            "object": "Spain"
        },
        {
            "subject": "Midfielder",
            "relation": "subclass of",
            "object": "Football player"
        },
        {
            "subject": "Midfielder",
            "relation": "instance of",
            "object": "Association football positions"
        },
    ]
</example>
```

Examples for the second step prompting

```
<example>
Input:
    Text: "Tahiti Honey" is an English-language film written by Frederick Kohner.

    Triplets and corresponding entity and relation mappings:

        {"subject": "Tahiti Honey", "relation": "language of film", "object": "English language"}
        {"Tahiti Honey": ["Tahiti Honey", "Honey", "Honey Chile", "Celtic Honey", "Tahitipresse"]}
        {"language of film": ["original language of film or TV show", "language regulatory body", "language used",
        "native language", "dialect of"]}
        {"English language": ["English language", "English Hours", "English International School", "British English",
        "Business English"]}

        {"subject": "Tahiti Honey", "relation": "is", "object": "film"}
        {"Tahiti Honey": ["Tahiti Honey", "Honey", "Honey Chile", "Celtic Honey", "Tahitipresse"]}
        {"is": ["represents", "instance of", "replaces", "contains", "is a list of"]}
        {"film": ["Film", "The Film", "Dance film", "Film award", "Romance Film"]}

        {"subject": "Tahiti Honey", "relation": "written by", "object": "Frederick Kohner"}
        {"Tahiti Honey": ["Tahiti Honey", "Honey", "Honey Chile", "Celtic Honey", "Tahitipresse"]}
        {"written by": ["lyrics by", "has lyrics", "produced by", "adapted by", "screenwriter"]}
        {"Frederick Kohner": ["Frederick Kohner", "Paul Kohner", "Adolf Kohner", "Susan Kohner", "Henry Rohner"]}


Output:
        [
                {
                        "subject": "Tahiti Honey",
                        "relation": "original language of film or TV show",
                        "object": "English language"
                },
                {
                        "subject": "Tahiti Honey",
                        "relation": "instance of",
                        "object": "Film"
                },
                {
                        "subject": "Tahiti Honey",
                        "relation": "screenwriter",
                        "object": "Frederick Kohner"
                }
        ]
</example>

<example>
Input:
    Text: ArXiv is funded by Los Alamos National Laboratory and Nagoya University.

    Triplets and corresponding entity and relation mappings:
        {"subject": "ArXiv", "relation": "funded by", "object": "Los Alamos National Laboratory"}
        {"ArXiv": ["ArXiv", "EngrXiv", "Physics arXiv Blog", "SocArXiv", "PsyArXiv"]}
        {"funded by": ["supervised by", "grants", "donated by", "endorsed by", "funder"]}
        {"Los Alamos National Laboratory": ["Los Alamos National Laboratory", "Los \u00c1lamos",
        "Los Alamos Technical Associates", "Los Alamos, California", "Los Alamos Neutron Science Center"]}

        {"subject": "ArXiv", "relation": "funded by", "object": "Nagoya University"}
        {"ArXiv": ["ArXiv", "EngrXiv", "Physics arXiv Blog", "SocArXiv", "PsyArXiv"]}
        {"funded by": ["supervised by", "grants", "donated by", "endorsed by", "funder"]}
        {"Nagoya University": ["Nagoya University", "Nagoya City University", "Nagoya Gakuin University",
        "Nagoya Institute of Technology", "Nagoya International School"]}

Output:
        [
                {
                        "subject": "ArXiv",
                        "relation": "funder",
                        "object": "Los Alamos National Laboratory"
                },
                {
                        "subject": "ArXiv",
                        "relation": "funder",
                        "object": "Nagoya University"
                },
        ]
</example>
```

Examples for the second step prompting

```
<example>
Input:
        Text: Lexington Avenue is a thoroughfare in New York City, parallel to Park Avenue and Third Avenue.

    Triplets and corresponding entity and relation mappings:

        {"subject": "Lexington Avenue", "relation": "is", "object": "thoroughfare"}
        {"Lexington Avenue": ["Lexington Avenue", "450 Lexington Avenue", "599 Lexington Avenue", "731 Lexington Avenue", "Lexington Park"]}
        {"is": ["represents", "instance of", "replaces", "contains", "is a list of"]}
        {"thoroughfare": ["Thoroughfare", "No Thoroughfare", "Thoroughfare Gap", "Parisian Thoroughfare", "Central Arc Thoroughfare"]}

        {"subject": "Lexington Avenue", "relation": "parallel to", "object": "Park Avenue"}
        {"Lexington Avenue": ["Lexington Avenue", "450 Lexington Avenue", "599 Lexington Avenue", "731 Lexington Avenue", "Lexington Park"]}
        {"parallel to": ["parallel street", "direction", "crosses", "adjacent station", "relative"]}
        {"Park Avenue": ["Park Avenue", "79 Park Avenue", "7 Park Avenue", "Park Avenue Plaza", "245 Park Avenue"]}

        {"subject": "Lexington Avenue", "relation": "parallel to", "object": "Third Avenue"}
        {"Lexington Avenue": ["Lexington Avenue", "450 Lexington Avenue", "599 Lexington Avenue", "731 Lexington Avenue", "Lexington Park"]}
        {"parallel to": ["parallel street", "direction", "crosses", "adjacent station", "relative"]}
        {"Third Avenue": ["Third Avenue", "Third Avenue Railway", "T Third Street", "Third Street Promenade", "1111 Third Avenue"]}


Output:
        [
                {
                        "subject": "Lexington Avenue",
                        "relation": "instance of",
                        "object": "Thoroughfare"
                },
                {
                        "subject": "Lexington Avenue",
                        "relation": "parallel street",
                        "object": "Park Avenue"
                },
                {
                        "subject": "Lexington Avenue",
                        "relation": "parallel street",
                        "object": "Third Avenue"
                },
        ]
</example>
```

B Ontology queries

As the number of relations in the Wikidata is relatively small compared to the number of entities (in order of a thousand) we scraped relation constraints (constraint classes from Q21503250 and Q21510865) in advance. Therefore, we decreased the number of calls to WDQS and increased the throughput of the pipeline.

In turn, to retrieve the subclass hierarchy of a specific entity, the following SPARQL query was used:

```
SELECT DISTINCT ?subclass ?subclassLabel WHERE {
            {
                wd:{entity_id} p:P31/ps:P31/wdt:P279* ?subclass.
            }
              UNION
            {
                wd:{entity_id} p:P279/ps:P279/wdt:P279* ?subclass.
            }
            SERVICE wikibase:label
            {bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en".}
```

Listing 1: SPARQL query for retrieving entity subclass hierarchy

The query identifies 'subclass' and 'instance of' values for a specific entity and goes up to the root of its hierarchy collecting all names representing the classes of entity of interest.

C Examples inconsistencies in synthetic dataset

SynthIE dataset contains frequent cases of discrepancy between generated text (aimed as input text for KG construction methods) and triplets (aimed as target outputs expected in KG construction. Such samples

introduce bias through LLM training and LLM in-context demonstration, resulting in skewed quality estimation for both methods.

Examples of mismatches between text and annotation in SynthIE

```
{
    "text": "Medan Marelan is in the same time zone as Krasnoyarsk.",
     "triplets": [
            ('Medan Marelan', "located in time zone",  "UTC+07:00'),
            ('UTC+07:00", "said to be the same as",
            "Krasnoyarsk Time')
    ]
}


{
    "text": "Dr. Babasaheb Ambedkar College of Arts, Commerce and Science is an academic
    institution in India.",

        "triplets": [
            ("Dr. Babasaheb Ambedkar College of Arts, Commerce and Science", "instance of",
            "Academic institution"),
            ("Dr. Babasaheb Ambedkar College of Arts Commerce and Science", "country",
            "India"),
            ("Academic institution", "subclass of", "Educational institution"),
            ("Academic institution", "subclass of", "Research institute"),
            ("India", "language used", "Brokskat"),
            ("India", "public holiday", "Independency Day (India)")
        ]
}
```

D Natural language dataset

We aimed to evaluate the performance across multiple domains, including those different from SynthIE-text-small. For this purpose, we manually sampled 100 paragraphs from Wikipedia that were relevant to WikicIE-test-small. We applied both our approach and SynthIE to the obtained text and compared the conformance of the resulting triplets with the input text. This sampling strategy ensures a fair comparison of models on a distribution that diverges from their pre-training data but remains within the same knowledge graph area.

For all model outputs, we selected the triplets that were consistent with the input text. These were either explicitly mentioned, implied, or readily derivable from the text without additional knowledge. This methodology aimed to evaluate the quality of triplet extraction in contexts closely resembling real-world applications, where the model meets unknown text mentioning known entities.

Each triplet was independently labeled by the three authors of this paper. A predicted triplet was deemed correct (i.e., consistent with the input text) if at least two of the three authors reached a consensus on its correctness. Precision was computed as the total number of valid triplets divided by the total number of predicted triplets across all samples.

The evaluation results show that both models perform well, but the proposed approach demonstrates much higher precision, though the number of generated triplets is close. It is worth noting that the predicted triplets of the two models differ from each other significantly, with the IoU metric less than 9%, meaning that less than 9 percent of correct triplets were predicted by both models.

Below we provide example pairs of original and natural language texts, each of both are about the same entity. Outputs of SynthIE T5-large and our method are highlighted with green color in case they are consistent with the input text and with red color otherwise:

Examples from the composed natural language dataset

Original text from the SynthIE dataset:
Elvira Barbey was the partner of Louis Barbey in business and sport. She competed in pair skating and participated in figure skating at the 1928 Winter Olympics – Pairs.

Natural language text from Wikipedia paragraph:
Elvira Barbey was a Swiss figure skater. She competed at the 1928 Winter Olympics and finished 19th in singles and 11th in pairs, together with her husband Louis Barbey.

SynthIE T5-large generated triplets:
(Elvira Barbey, sport, Figure skating)
(Elvira Barbey, country for sport, Switzerland)
(Elvira Barbey, participant in, 1928 Winter Olympics)
(Elvira Barbey, spouse, Louis Barbey)

GPT generated triplets:
(Elvira Barbey, country of citizenship, Switzerland)
(Elvira Barbey, participant in, 1928 Winter Olympics)
(1928 Winter Olympics, instance of, Olympic Games)
(Elvira Barbey, partner in business or sport, Louis Barbey)
(Louis Barbey, partner in business or sport, Elvira Barbey)

Original text from the SynthIE dataset:
Ilya Shatrov was a military personnel who served during the Russo-Japanese War. He was also a trumpet player.

Natural language text from Wikipedia paragraph:
Ilya Alekseevich Shatrov was a Russian military musician, conductor and composer, known for composing the waltz On the Hills of Manchuria in 1906, recounting his experiences at the Battle of Mukden during the Russo-Japanese War, which he dedicated to one of his comrades fallen at the battle.

SynthIE T5-large generated triplets:
(On the Hills of Manchuria, form of creative work, Waltz (music))
(On the Hills of Manchuria, language of work or name, Russian language)
(On the Hills of Manchuria, copyright status, Public domain)
(On the Hills of Manchuria, composer, Ilya Averbakh)
(Ilya Averbakh, conflict, Battle of Mukden)
(Ilya Averbakh, genre, Opera)
(Ilya Averbakh, conflict, Russo-Japanese War)
(Ilya Averbakh, notable work, On the Hills of Manchuria)

GPT generated triplets:
(On the Hills of Manchuria, publication date, 1906)
(Battle of Mukden, described by source, On the Hills of Manchuria)
(Battle of Mukden, part of, Russo-Japanese War)

Original text from the SynthIE dataset:
Srđan Radosavljev is a member of the Serbian League Vojvodina team FK Palić, where he plays as a midfielder.

Natural language text from Wikipedia paragraph:
Srđan Radosavljev is a Serbian professional footballer who plays mainly as midfielder for FK Bačka 1901. Born in Vršac, SR Serbia, he represented several clubs during his career, spending most of the time with Mladost Apatin and Banat Zrenjanin.

SynthIE T5-large generated triplets:
(UTC+01:00, said to be the same as, Central European Time)
(Ankara, located in time zone, Central European Time)
(Ankara, located in time zone, UTC+02:00)
(Ankara, located in time zone, UTC+01:00)

GPT generated triplets:
(Srđan Radosavljev, position played on team / speciality, Midfielder)
(Srđan Radosavljev, member of sports team, FK Bačka 1901)
(Srđan Radosavljev, sport, Association football)
(Srđan Radosavljev, place of birth Vršac)
(Srđan Radosavljev, country of citizenship, Serbia)
(Srđan Radosavljev, languages spoken, written or signed, Serbian language)