Ontology-guided Knowledge Graph Construction from Maintenance Short Texts

Zeno van Cauter
Technische Universiteit Eindhoven
z.m.v.cauter@tue.nl

Nikolay Yakovets
Technische Universiteit Eindhoven
n.yakovets@tue.nl

Abstract

Large-scale knowledge graph construction remains infeasible since it requires significant human-expert involvement. Further complications arise when building graphs from domain-specific data due to their unique vocabularies and associated contexts. In this work, we demonstrate the ability of open-source large language models (LLMs), such as Llama-2 and Llama-3, to extract facts from domain-specific Maintenance Short Texts (MSTs). We employ an approach which combines ontology-guided triplet extraction and in-context learning. By using only 20 semantically similar examples with the Llama-3-70B-Instruct model, we achieve performance comparable to previous methods that relied on fine-tuning techniques like SpERT and REBEL. This indicates that domain-specific fact extraction can be accomplished through inference alone, requiring minimal labeled data. This opens up possibilities for effective and efficient semi-automated knowledge graph construction for domain-specific data.

1 Introduction

Knowledge Graphs (KGs) have emerged as a powerful tool for representing complex relationships between entities across various domains and in aiding in various tasks (e.g., in search, recommendation systems, and others) (Hogan et al., 2021).

Constructing a KG presents several challenges. The process requires extracting structured information from unstructured data, such as text, using Information Extraction (IE) techniques. Much research has focused on large, publicly available general-purpose KGs like DBPedia, YAGO, or Wikidata, as well as on domain-specific KGs in areas like medicine (Li et al., 2020) or railway safety (Liu et al., 2021). More recent studies have explored the use of KGs to support industrial maintenance activities (Hossayni et al., 2020; Stewart et al., 2022). However, building a maintenance KG involves overcoming several additional obstacles: off-the-shelf Natural Language Processing solutions often fail to handle domain-specific data adequately, existing benchmarks do not align with industrial realities, the costs of annotating domain-specific data can be prohibitive, and the typically low volume of domain-specific data makes it challenging to train robust models that generalize well to new instances (Brundage et al., 2021; Dima et al., 2021). Additional difficulties arise when data evolves (e.g., triggering changes in the label space) necessitating computationally-expensive retraining or fine-tuning of models in traditional approaches.

In-context-learning (Dong et al., 2022) and ontology-guided KG construction from Text2KGBench (Mihindukulasooriya et al., 2023) offer the ability to overcome some of these challenges. Both these methods are dynamic and adaptable to changes in the ontology or label space without the need for re-training. In-context learning does not require large collection of annotated labeled data upfront but only at time of inference. Ontology-guided KG construction allows for seamless changes to the ontology if desired. This makes these methods particularly useful in domains where ontologies evolve over time.

Recently, Large Language Models (LLMs) have demonstrated remarkable capabilities in the ability to perform information extraction (Xu et al., 2023). However, most of this work focuses on general domain datasets, e.g. ACE datasets [1][2], CoNLL2003 (Tjong Kim Sang and De Meulder, 2003) or TacRED (Zhang et al., 2017) and little work exists on specialized domain-specific datasets. An annotated dataset of fine-grained schema and corpora for information extraction of Maintenance Short Texts (MST) recently became publicly available:

[1] https://catalog.ldc.upenn.edu/LDC2005T09
[2] https://catalog.ldc.upenn.edu/LDC2006T06

Proceedings of the 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024), pages 75–84

MaintIE (Bikaun et al., 2024a).

In this work, we show how LLMs can assist with the knowledge graph construction on domain-specific texts. Our contributions are as follows:

1. We evaluate the LLama-2 (Touvron et al., 2023a) and LLama-3[3] family models on ontology-guided KG construction using in-context-learning on a dataset of Maintenance Short Texts (Bikaun et al., 2024a).

2. We show that, using only a few in-context examples, Llama-3-70B-Instruct can extract fact extracts comparable to previous state-of-the-art with a near-zero hallucination rate. We find that for other models of the Llama-family, hallucinations come into play where generated triples contain objects/subjects from (mostly) in-context examples.

3. We study the effects of choosing certain token prediction penalties and the effects on hallucinations. We show that by carefully selecting these parameters can minimize the number of hallucinations, but that the wrong settings can stimulate this behaviour.

4. Finally, we show that the pruning of such hallucinations is relatively easy and increases performance (in both precision and F1) by a large margin. Performing this pruning makes smaller models such as Llama-3-8B a suitable alternative.

Our work implies that LLMs are well-suited for building domain-specific knowledge graphs, even with limited supervised data. In addition, if large-scale data annotation is required, LLMs can be combined with a human-in-the-loop process that pre-annotates data at an incremental rate. Our code, prompts and data are publicly available[4].

2 Task description

In this work, we consider the task of LLM-assisted KG construction as automatically extracting graph structured information (subject, object and (directional) relation) from unstructured text data. In line with Text2KGBench, we also regard this task as "Given an ontology and text corpora, the goal is to construct prompts to instruct the model to extract facts relevant to the ontology". An example of how this is setup in the prompt is given in Figure 1.

[3]https://ai.meta.com/blog/meta-llama-3/

[4]https://github.com/zeno17/MaintIE2KGBench

3 Methodology

3.1 Data

MaintIE (Bikaun et al., 2024a) provides a collection of Maintenance Short Texts (MST's) which encapsulates information from Maintenance Work Orders (MWOs) in a lexically-normalised concise format (Bikaun et al., 2024b). It comes in 2 annotation versions: 1) Fine-grained, spanning 224 entity classes or 2) Course-grained, spanning 6 entity classes. The fine-grained version is the result of pure intensive expert annotation, and the course-grained version was created by performing pre-annotation using fine-tuned SpERT (Eberts and Ulges, 2019) which was followed by expert correction. An example text with corresponding triplets is provided below.

```
Text:
cabin lights require replacing
Ground truth triples:
hasPart(cabin,lights)
hasAgent(require,lights)
hasPatient(require,replacing)
```

As both versions come with the same 6 relation types, we opt for the course-grained data as it is more numerous (7.000 compared to 1.067). From this, we filter out MST's that don't have actual triples annotated to them. This follows Text2KGBench which 1) also only uses triple-containing texts and 2) whose evaluation framework is not equipped to measure performance over non-triple containing texts. This only filters out 272 examples or 3.9% of the data.

From the remaining 6.728 examples, we create a 75/25 train-test split (or 5.046/1.682 examples respectively). During the experiments, the examples given to the model in the context are drawn from the train split, and performance is measured over the held-out test-split. More on this is covered in Subsection 3.5.

3.2 Prompt

For the prompting, we include a basic instruction, an ontology, k examples and the test sentence. The prompt template is provided in Figure 1. This differs from Text2KGBench as follows: 1) we feed multiple examples to model, and 2) we do not provide relation constraints to the model (which entities can have which relations). We do not provide the relation constraints as this takes a considerable amount of space in the context-window of the LLM.

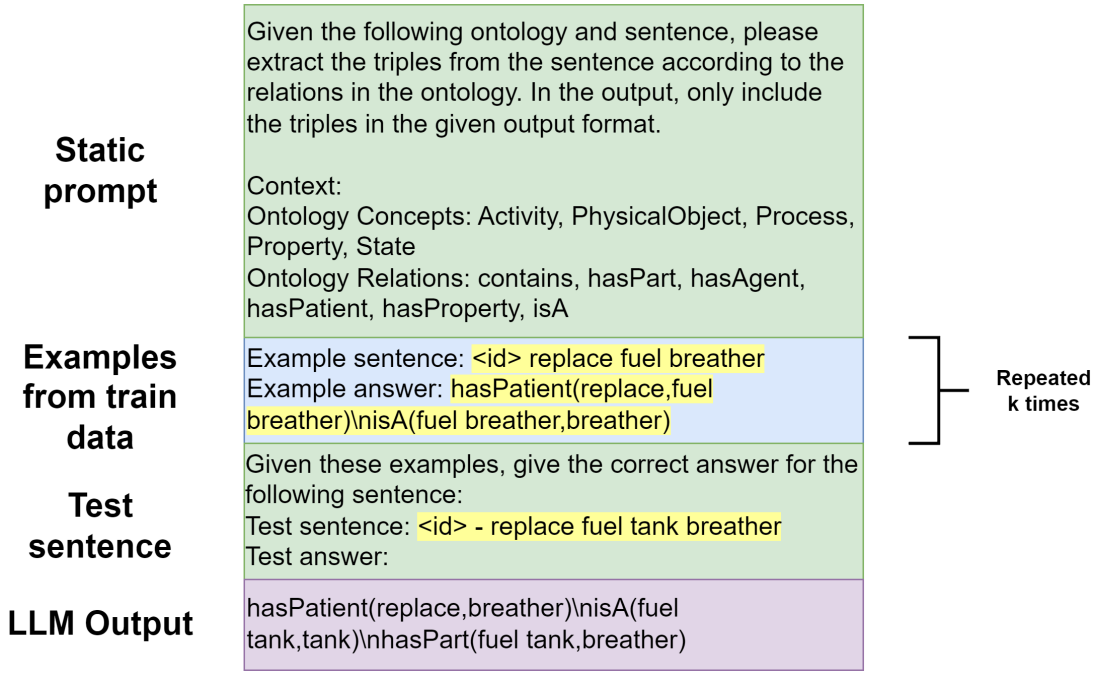

Figure 1: Used prompt template

While this space is limited for the course-grained data (5 entity types, 6 relations), the amount of space required can grow intractably for larger ontologies (e.g. the fine-grained dataset has 224 entity types and 6 relations). We consider this an avenue for future work.

3.3 Metrics

For evaluation Text2KGBench focuses on three dimensions: 1) fact extraction performance 2) ontology conformance, and 3) hallucination rate. Below, we provide brief explanations of the evaluation metrics, where we deviate from them and why.

1. **Fact Extraction**: From the generated text, triplets of the form "relation(subject, object)" are extracted using regular expressions. The extracted triples are then compared to the set of ground truth triples, and performance is measured using Precision, Recall and F1-score. Any triple that is not an exact match for relation type, object or subject is considered incorrect.

2. **Ontology Conformance**: Is the predicted relation in the provided ontology (provided in Figure 1). In this work, we limit ourselves to the relations: ['contains', 'hasPart', 'hasAgent', 'hasPatient', 'hasProperty', 'isA'].

3. **Hallucination rate**: Whether the LLM predicts relations that are not in the ontology, or objects/subjects that are not in the provided text. As Text2KGBench introduces two benchmark datasets based on Wikidata and DBpedia. This data carries more linguistically variation for the entities, and therefore they use a loose regime where objects/subjects are matched through stemmed words (using the Porter stemming algorithm (Van Rijsbergen et al., 1980)). In our work, we only count exact matches as correct because the MaintIE data is of limited vocabulary variation. We only consider exact matches and anything outside of that we consider a hallucination. For example, if the word "filter" is in the target sentence, a triple containing "filters" as an object/subject is considered a hallucination. This is important in a maintenance setting as, for example, having a singular or multiple component carries different semantics or may not even possible (e.g. if a machine only has the component once).

3.4 Models

LLMs are neural-inspired models that are trained on immense amounts of data. While initially designed for machine translation (Vaswani et al., 2017), adaptations such as encoder-only BERT (Devlin et al., 2018) or decoder-only GPT (Radford

and Narasimhan, 2018) found use for a plethora of tasks. Recently, GPT-based models have been found to be the most versatile and flexible through its generative nature, including for generative information extraction (Xu et al., 2023).

LLaMa (Touvron et al., 2023a,b)[5], is an open-source LLM, and comes in different sizes and both only pre-trained and instruction-tuned versions.

In this work, we will assess several releases of the Llama family and assess their capabilities of performing fact extraction in the maintenance domain. We consider the following versions:

1. Llama-2-7B[6]
2. Llama-2-70B[7]
3. Llama-3-8B[8]
4. Llama-3-8B-Instruct[9]
5. Llama-3-70B[10]
6. Llama-3-70B-Instruct[11]

3.5 In-Context Learning

In-context-learning (ICL) is a technique of providing an LLM with a few examples to create a demonstration context. It then combines a query question with this context to form a prompt, which is fed into a language model for prediction. The model is expected to discern the pattern in the demonstration and make the appropriate prediction (Dong et al., 2022).

The model's context length is a hard limit on how many examples can be used, and the number of examples that necessary or effective can differ per model. In the context of maintenance data, availability is an important bottleneck as human annotated data is time-consuming and expensive. For this reason, we will experiment how many examples the model needs to be provided with in the context to do an effective fact extraction. For every example in the test set, semantically similar examples are retrieved using *sentence-transformers*[12] (Reimers and Gurevych, 2019) and the all-mpnet-base-v2 model[13]. In Text2KGBench, the models are only provided a single example (k=1). In our case, we will experiment with k $\in \{1, 2, 3, 5, 10, 20, 50, 100, 150\}$ except for Llama-2 where 100 and 150 examples are not possible due to hitting the context length limit. Still, this is a high number of examples which is possible largely because the maintenance short text data is of limited length.

3.6 Token prediction penalties

Text2KGBench demonstrated that ontology-guided information extraction suffers from hallucinations. This means triples are generated where the relation does not conform to the ontology or where subjects and objects that were not in the test sentence in the first place. During early experimentation, we found that the used LLM's tend to do (among others) the following: 1) repeat the same tokens until maximum sequence length was reached, and 2) provide lengthy explanations despite only asking for triples, including the generation of code.

For our LLM implementation, the parameters "frequency penalty" and "presence penalty" can be used. These change the logits if the LLM uses same tokens repeatedly or encourages it to use different tokens than already seen. Using Llama-3-8B (for computational reasons) we experiment with different settings in the full available range ([-2, 2]) to see how restricting the output logits affects the LLMs performance. As ontology conformance is generally high (and thus relation hallucination rate low), we look at the averaged hallucination rate of the object and subject. From our preliminary findings, we decided to run all other experiments with a frequency penalty of 0 and a presence penalty of -1.

3.7 Hallucination types

Next, we inspect some intricacies of the hallucinations that we found. We select the predictions from Llama-3-8B on 10 examples with frequency penalty 0, and presence penalty 2, which has the highest combined subject-object hallucination rate in our work (0.22 and 0.21 respectively). However, a solid inspection framework grows fast in complexity considering hallucination intricacies, let alone proving direct causality. We scope our approach in order to provide some quantitative inspection, and leave a hallucination inspection framework for future work. In the end, we limit ourselves to the following:

1. We only expand upon subject/object halluci-

[5] https://ai.meta.com/blog/meta-llama-3/
[6] https://huggingface.co/meta-llama/Llama-2-7b
[7] https://huggingface.co/meta-llama/Llama-2-70b
[8] https://huggingface.co/meta-llama/Meta-Llama-3-8B
[9] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
[10] https://huggingface.co/meta-llama/Meta-Llama-3-70B
[11] https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct
[12] https://github.com/UKPLab/sentence-transformers
[13] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

nations for simplicity.

2. We only consider subjects consisting of 1 word (e.g. "replace" but not "change out" or "chain hoist) due to difficulties with proper stemming.

3. We adopt an assumed hierarchy of errors of most probable cause of the hallucination, which is as follows (in order):
   (a) The stemmed subject/object is in a stemmed sentence (this matches e.g. replace and replacing).
   (b) The subject/object is in one of the examples provided to the LLM.
   (c) The stemmed subject/object is in one of the stemmed examples.

3.8 Hallucination-filtered performance

Lastly, we look at what the fact extraction performance can be without hallucinations. If a triple contains a relation not in the provided ontology, or a subject/object which is not in the original input string, then it simply cannot be a factual triple. These conditions can be verified automatically and triples that violate them filtered from the fact extraction process. This leads to less produced triples, but the remaining extracted triples should match better with the ground truth and thus increase precision.

4 Results

4.1 Fact Extraction

Figure 2 shows how effective each LLM is at obtaining correct facts and hallucination rate versus the number of examples. It can be seen that there are stark differences between model performance with both highest and lowest performance coming from the instruction-tuned and untuned Llama-3-70B respectively. Conversely, for Llama-3-8B instruction-tuning seems to decrease performance across the board. In addition, Llama-3-8B-Instruct has a visibly lower ontology conformance compared to the other models which all adhere to the provided ontology systematically. Eventually, Llama-3-70B-Instruct obtains 0.77 F1-score when given 150 examples. For both versions of Llama-3-8B, further increasing the number of examples to a 150 hurts performance compared to fewer examples. The scores of k=20 (which we consider a low amount) are also displayed in Table 1.

4.2 Token prediction penalties

Figure 3 shows how Llama-3-8B's performance varies when tuning different parameters as described in Subsection 3.6. It can be seen that shifting frequency penalty and token penalty leads to an optimal fact extraction performance on an off-diagonal line. In addition, the lower right triangle is the generally lower performing side in terms of fact extraction. Conversely, this lower performance is combined with an increasing hallucination rate.

4.3 Ontology conformance & Hallucination rate

Text2KGBench reports that ontology conformance is consistently high across a variety of ontologies, which resonates with our results. In Figure 2 the ontology conformance is near 1 for all models, with the exception to this are Llama-2-7b and Llama-3-8B-Instruct where we see a decline throughout the number of examples. This means that the LLM's generally adhere to the provided Ontology, at a much higher rate found for the Text2KGBench benchmark.

Next, we look at how performance and hallucination progresses as the number of in-context examples increases in Figure 4. For Llama-3-8B-Instruct, the hallucination rate first increases follow by stabilization. Both Llama-3-8B and Llama-3-8B-Instruct suffer from a the hallucination rate and this is relatively stable as the number of examples increases. On the contrary, Llama-3-70B-Instruct does not suffer from this problem and sees a steady performance increase while the hallucination rate actually goes down. Thus, this seems to be a model-dependent issue.

4.4 Hallucination types

From inspection, we found that most subject/object hallucinations conform to the following scenarios: 1) objects/subjects contain tokens from the examples provided in the context, 2) objects/subjects being changed from plural to singular or vice versa, 3) object/subject verbs having active instead or passive form or vice versa. In some cases, these observations are not mutually exclusive for a single sentence. For example: if a test sentence contains "replaced", an extracted triple has a subject 'replace', and the word "replace" occurs in an example, then both 1) and 3) are true simultaneously. For relation hallucinations, the LLM sometimes used the provided ontology concepts as a relation (e.g. Phys-

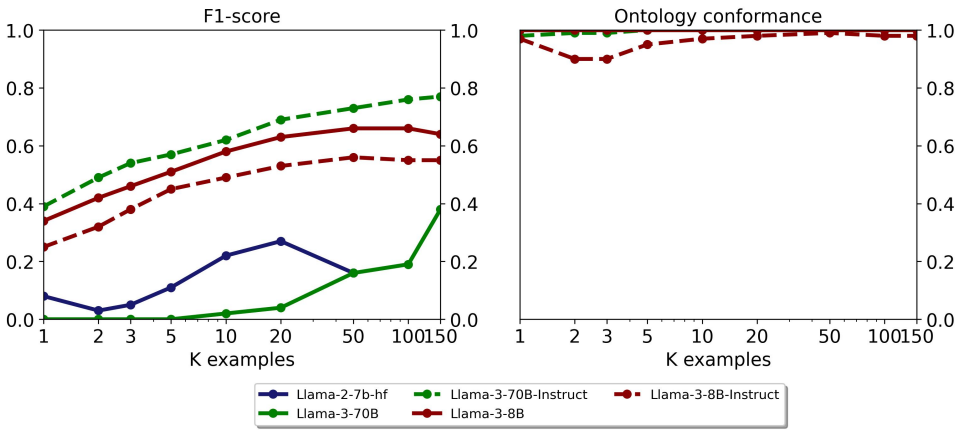

Figure 2: Left: Per-model performance on fact extraction. Right: Ontology conformance. Higher is better. Scale is logarithmic

Model	P (↑)	R (↑)	F1 (↑)	OC (↑)	SH (↓)	RH (↓)	OH (↓)
REBEL (MaintIE)	-	-	0.77	-	-	-	-
Llama-2-7b-hf	0.31	0.26	0.27	1.00	0.03	0.00	0.01
Llama-3-8B	0.62	0.70	0.63	1.00	0.03	0.00	0.03
Llama-3-8B-Instruct	0.48	0.70	0.53	0.98	0.08	0.02	0.09
Llama-3-70B	0.04	0.04	0.04	1.00	0.00	0.00	0.00
Llama-3-70B-Instruct	0.67	0.74	0.69	1.00	0.00	0.00	0.01

Table 1: Per-Model Fact Extraction Performance, Ontology Conformance and Hallucination rate. Scores reported are Precision, Recall, F1-score, Ontology Conformance, Subject Hallucination, Relation Hallucination and Objection Hallucination. Number of examples (k) is 20. For P, R, F1 and OC higher is better (↑). For SH, RH and OH lower is better (↓).

icalObject, Process, etc.), or it combined them into new relations (e.g. the relation hasProcess from the concept Process, hasState from State, etc.). It also occured the generated answer contained Python code (despite being asked not to) where certain lines contained substrings matching a "r(a,b)" form which were extracted unintentionally.

Figure 5 partially quantifies some of these aspects and it can be seen that for both subject and both, a large part of hallucinations overlap with being present in the context examples.

4.4.1 Hallucination-filtered performance

If triples that contain a hallucinated relation, object or subject are pruned, we obtain the performance as reported in Table 2. We observe that by applying a simple filter for triples of which we know they are non-factual, all models gain a significant amount of performance. The exception being Llama-3-70B-Instruct as it already obtained high performance with near-zero hallucination rate. We observe that for all models, the precision improves (as expected) compared to the results in Table 1. This heuristic pruning of extracted triples can thus be a useful way of increasing fact extraction performance, specifically smaller models which require less compute power.

5 Discussion

Firstly, we will draw a comparison to the results of MaintIE (Bikaun et al., 2024a). Since we only focus on triplet extraction without entity recognition, a comparison must be done between our work and MaintIE's evaluation of REBEL on loose relation extraction (as it only requires agreement on the relation type and entity spans) (Bikaun et al., 2024a). In a supervised fine-tuning setting called curriculum learning, MaintIE (Bikaun et al., 2024a) obtained an F1-score of 0.77. For comparison, Llama-3-70B-Instruct matches this score by using 150 examples and obtains 0.69 F1-score using only 20 semantically similar in-context examples, making the performance remarkably close. The effectiveness of using only a few semantically similar examples can significantly improve the model's

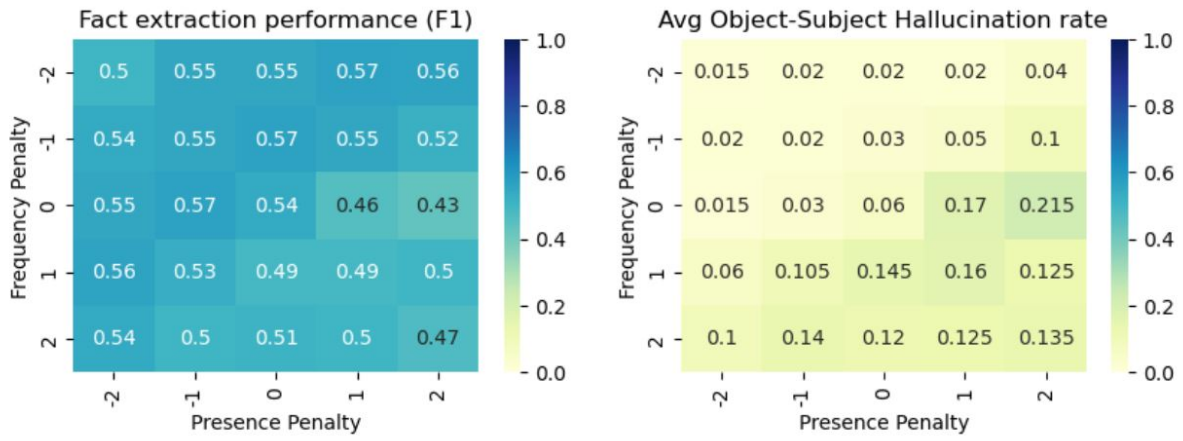

Figure 3: Fact extraction performance and hallucination rate for different settings of frequency and presence penalties. Selected model was Llama-3-8B. Number of in-context examples was set to 10. Left: higher is better. Right: lower is better.

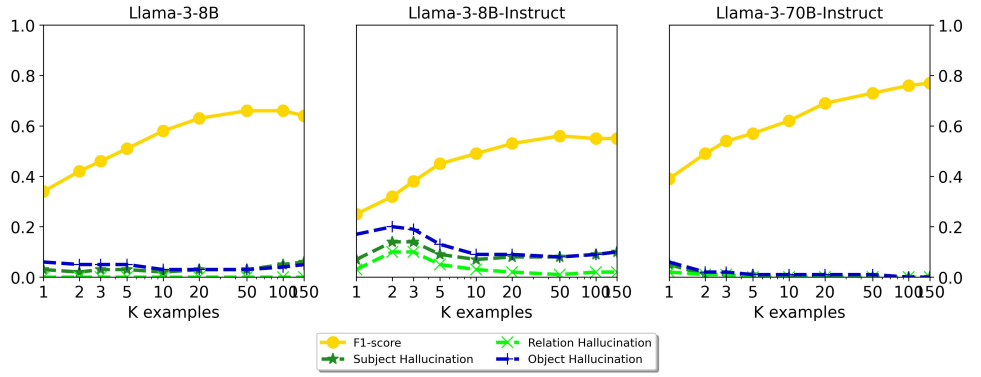

Figure 4: Number of in-context examples versus hallucination rate. Llama-3-8B, Llama-3-8B-Instruct and Llama-3-70B-Instruct selected for their overall performance.

ability to recognize patterns in the data.

However, this performance is only close when comparing it to the largest state-of-the-art open and instruction-tuned models. For Llama-3-70B, its low performance is explained that a significant portion of its "generations" are empty, which means its low performance is caused by the model's failure to even generate a sequence with triples at all. The associated performance in terms of hallucination is therefore void, given that empty generations by default don't contain tokens that fall outside the given sentence of the ontology. Llama-3-70B without instruction-tuning is thus incapable of performing fact extraction, while instruction-tuning Llama-3-8B slightly decreases performance rather than improve it.

Second, we would like to draw a comparison methodologically between REBEL, SpERT and LLMs and review differences and corresponding consequences. Both REBEL and SpERT use a fine-tuning approach that requires the labelled data to be available upfront. For SpERT, a relation classifier is used and as it constrains its output to a label space, it doesn't suffer from hallucinations. LLMs do not require this labelled data for fine-tuning, and, in this work, we have shown that even with few examples they can already be effective. However, this at-inference requirement of LLM comes with the drawback of hallucinations and is a subject of research (McKenna et al., 2023; Agrawal et al., 2023).

Thirdly, we will discuss how these hallucinations can be dealt with. We find that changing token penalties can simultaneously maximize fact extraction performance and minimize hallucination rate. By stimulating the model to diversify through presence penalty, the generated hallucinated triples will contain objects/subjects that are outside of the target sentence, likely sourced by in-context-examples. The exact reason for why

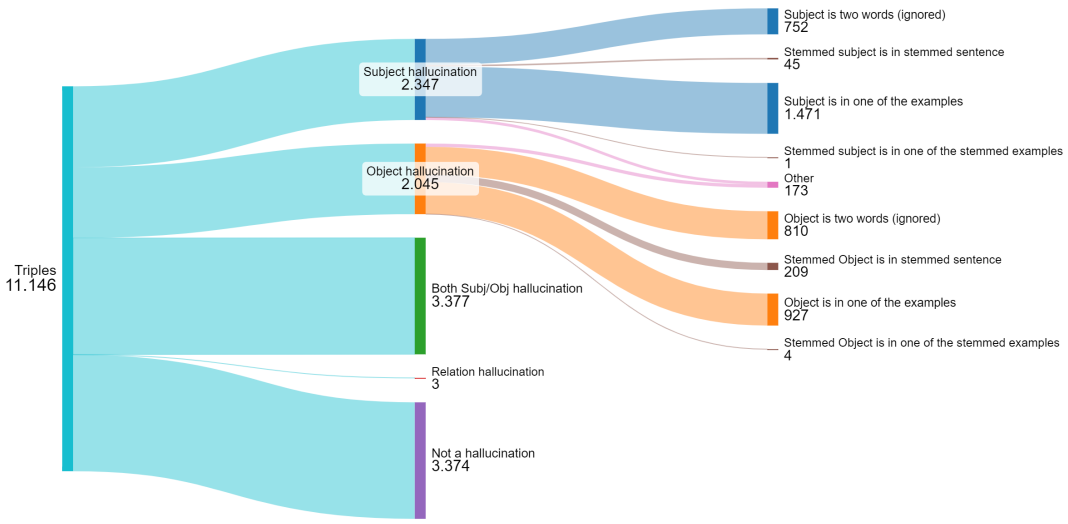

Figure 5: Types of hallucinations and subject sub-classifications. Based on predictions from Llama-3-8B with frequency penalty 0 and presence penalty 2. These parameter values induce a relatively high number of hallucinations.

	Before pruning			After pruning		
Model	P	R	F1	P	R	F1
REBEL (MaintIE)	-	-	0.77	-	-	-
Llama-2-7b-hf	0.31	0.26	0.27	0.32	0.26	0.28
Llama-3-8B	0.62	0.70	0.63	0.64	0.70	0.65
Llama-3-8B-Instruct	0.48	0.70	0.53	0.58	0.69	0.61
Llama-3-70B	0.04	0.04	0.04	0.04	0.04	0.04
Llama-3-70B-Instruct	0.67	0.74	0.69	0.68	0.74	0.69

Table 2: Fact extraction performance where hallucinations are pruned. Scores reported are Precision, Recall, F1-score where higher is better. Number of examples (k) is 20.

this occurs is unclear, and we consider an extended evaluation framework an interesting area for further research. Despite these hallucinations occurring, it is relatively straight-forward to prune them. Hallucinations are fairly easy to detect in this setting, as the relation must conform to the provided ontology and the subject/object must occur in the target text. The filtering of these verifiable hallucinations generally leads to a higher precision and thus higher F1-score, while ensuring ontology conformity in a domain-specific setting.

Lastly, we would like to discuss the implications of our findings. Building domain-specific Knowledge Graphs is a time-consuming effort, and building NLP-pipelines to do this often requires considerable resources. Our work implies that an incremental human-in-the-loop process could significantly assist with fact extraction. In (Bikaun et al., 2024a), pre-annotation was done by fully fine-tuning SpERT on an already annotated corpus and annotating a second corpus. Our work implies that by using LLMs and in-context learning, pre-annotation could start both earlier (using few examples) and continuously (building the number of examples as you go) using inference-only. This could considerably reduce workload for domain experts that need to be involved.

6 Conclusion

This study explores the use of Large Language Models for constructing knowledge graphs from Maintenance Short Texts. We assess models from the Llama family, focusing on fact extraction through two main methods: 1) ontology-guided triplet extraction and 2) in-context learning. Utilizing these techniques with the Llama-3-70B-Instruct model, we achieve fact extraction performance comparable to the current state-of-the-art methods that require fine-tuning. During this process, the issue of hallucinations (incorrect or fabricated information) can arise, often exacerbated by suboptimal

settings for token prediction penalties. However, for the Llama-3-70B-Instruct model, hallucinations are almost non-existent. For other models, it's feasible to prune hallucinated triples from the output. This capability extends even to smaller models like Llama-3-8B, making them viable alternatives. This approach facilitates human-in-the-loop pre-annotation for domain-specific datasets, potentially reducing the time investment required from domain experts. Our work shows that Large Language Models are a fitting solution for Knowledge Graph construction, specifically where labelled data is scarce or the ontology dynamic.

Acknowledgments

This work was made possible by the TKI MATTER grant. We also would like to thank Mykola Pechenizkiy, Tyler Bikaun and Simon Koop for their comments.

References

Garima Agrawal, Tharindu Kumarage, Zeyad Alghami, and Huan Liu. 2023. Can knowledge graphs reduce hallucinations in llms?: A survey. *arXiv preprint arXiv:2311.07914*.

Tyler Bikaun, Tim French, Michael Stewart, Wei Liu, and Melinda Hodkiewicz. 2024a. Maintie: A fine-grained annotation schema and benchmark for information extraction from maintenance short texts.

Tyler Bikaun, Melinda Hodkiewicz, and Wei Liu. 2024b. MaintNorm: A corpus and benchmark model for lexical normalisation and masking of industrial maintenance short text. In *Proceedings of the Ninth Workshop on Noisy and User-generated Text (W-NUT 2024)*, pages 68–78, San Ġiljan, Malta. Association for Computational Linguistics.

Michael P. Brundage, Thurston Sexton, Melinda Hodkiewicz, Alden Dima, and Sarah Lukens. 2021. Technical language processing: Unlocking maintenance knowledge. *Manufacturing Letters*, 27:42–46.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alden Dima, Sarah Lukens, Melinda Hodkiewicz, Thurston Sexton, and Michael P. Brundage. 2021. Adapting natural language processing for technical text. *Applied AI Letters*, 2(3):e33.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Markus Eberts and Adrian Ulges. 2019. Span-based joint entity and relation extraction with transformer pre-training. In *European Conference on Artificial Intelligence*.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4):1–37.

Hicham Hossayni, Imran Khan, Mohammad Aazam, Amin Taleghani-Isfahani, and Noel Crespi. 2020. Semkore: Improving machine maintenance in industrial iot with semantic knowledge graphs. *Applied Sciences*, 10(18).

Linfeng Li, Peng Wang, Jun Yan, Yao Wang, Simin Li, Jinpeng Jiang, Zhe Sun, Buzhou Tang, Tsung-Hui Chang, Shenghui Wang, and Yuting Liu. 2020. Real-world data medical knowledge graph: construction and applications. *Artificial Intelligence in Medicine*, 103:101817.

Jintao Liu, Felix Schmid, Keping Li, and Wei Zheng. 2021. A knowledge graph-based approach for exploring railway operational accidents. *Reliability Engineering & System Safety*, 207:107352.

Nick McKenna, Tianyi Li, Liang Cheng, Mohammad Hosseini, Mark Johnson, and Mark Steedman. 2023. Sources of hallucination by large language models on inference tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2758–2774, Singapore. Association for Computational Linguistics.

Nandana Mihindukulasooriya, Sanju Tiwari, Carlos F Enguix, and Kusum Lata. 2023. Text2kgbench: A benchmark for ontology-driven knowledge graph generation from text. In *International Semantic Web Conference*, pages 247–265. Springer.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing*.

Michael Stewart, Melinda Hodkiewicz, Wei Liu, and Tim French. 2022. Mwo2kg and echidna: Constructing and exploring knowledge graphs from maintenance data. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, page 1748006X2211311.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

C.J. Van Rijsbergen, S.E. Robertson, and M.F. Porter. 1980. *New Models in Probabilistic Information Retrieval*. British Library research & development reports. Computer Laboratory, University of Cambridge.

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

Derong Xu, Wei Chen, Wenjun Peng, Chao Zhang, Tong Xu, Xiangyu Zhao, Xian Wu, Yefeng Zheng, and Enhong Chen. 2023. Large language models for generative information extraction: A survey. *arXiv preprint arXiv:2312.17617*.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 35–45.