# Assessing the Evolution of LLM capabilities for Knowledge Graph Engineering in 2023

Johannes Frey[1][0000−0003−3127−0815], Lars-Peter Meyer[1][0000−0001−5260−5181], Felix Brei[1][0009−0008−5245−6655], Sabine Gründer-Fahrer[1][0000−0003−0054−5003], and Michael Martin[1,2][0000−0003−0762−8688]

[1] InfAI / Leipzig University, Germany, frey@infai.org, https://infai.org
[2] Chemnitz University of Technology, Germany, https://tu-chemnitz.de/

**Abstract.** In this article, we evaluate the evolution of LLM capabilities w.r.t. the RDF Turtle and SPARQL language as foundational skills to assist with various KGE tasks. We measure the LLM response quality using 6 LLM-KG-Bench tasks for a total of 15 LLM versions available over the course of 2023, covering 5 different "major version" LLM classes (GPT-3.5 Turbo, GPT-4, Claude 1.x, Claude 2.x, and Claude Instant 1.x).

**Keywords:** LLM benchmarking· knowledge graph engineering · RDF · large language models · evolution of llms

## 1 Introduction

Combining the power of large language models (LLMs) and knowledge graphs (KGs) [16,8,15] to improve both LLM response quality, but also automation of the creation or processing of KGs via LLM-assistants, received huge attention 2023. In order to assist with KG engineering (KGE), it is crucial that LLMs can access, understand, and manipulate KGs but also artifacts that are involved in their construction within various KGE processes. RDF Turtle is a widely adopted language for representing KGs and KGE artifacts (RML mappings, SHACL shapes, SPARQL BGPs, Ontologies, etc.) and could serve in combination with SPARQL as a general low-level KG(E) interface to be leveraged by LLMs. While there exist works that employ or study LLMs for KGC/KGE tasks, investigating the performance of LLMs w.r.t. such low-level interfaces and basic graph comprehension, still remains under-explored, albeit a studies showed [13,4] that syntactical issues hinder the usefulness of semantically meaningful responses. This study compared leading GPT4all models to leading commercial models w.r.t. how well they speak Turtle. However, the obtained results only represent a snapshot from July 2023, LLMs rapidly evolved over the course of 2023 and studies reported that the performance of newer LLM versions can decrease [9,3] for selected workloads, raising the question whether this also affects KGE workloads. In this paper, we focus on closing this gap by assessing the evolution in Turtle and SPARQL query generation skills of the commercial models from

that study, for all versions that were accessible at the end of December 2023. Our work encompasses the following novelties and contributions:

- We present the first comprehensive, quantitative, and qualitative examination of the evolution of Turtle and SPARQL language skills for all Claude and GPT-3.5T/GPT-4 releases (until Dec 2023). We assess 15 versions and provide evidence that the performance of specific successors decreased for a subset of RDF-KGE tasks.
- We published a reusable time capsule dataset [6], capturing experiment replay data enabling further in-depth investigations (e.g. custom scores). Given the (ongoing) discontinuation of old(er) OpenAI models, this resource is a valuable asset, since it also enables future LLM efforts to be compared with a state-of-the-art-2023-baseline independent of model availability.
- We released an evolved version [11] of the LLM-KG-Bench framework [12] with updated prompts (enhanced clarity), a novel SPARQL task, a feature to rerun (modified) evaluations on captured model responses (e.g. using the time capsule), and support for instantiation-based tasks.
- We performed a replication experiment of findings in [4], thereby verifying and reinforcing the original research outcomes and the soundness of the benchmark setup and tasks.

## 2 Related Work

Recent studies have started to explore the use of LLMs in the context of various KGE tasks. For instance, [22] investigated the performance of LLMs on typical KG construction (KGC) tasks, namely entity, relation, and event extraction as well as link prediction, on eight benchmark datasets. Similarly, [18] showed that current LLMs can be used to streamline the process of automatic creation of KGs from raw texts as well as for automatic ontology creation. In [7,20], LLMs have been used for automatic Knowledge Base construction, completion or correction. Utilizing instruction training on LLMs was motivated and showcased by [10] for RDF(S) triple generation from text, performing further RDF(s) reasoning and constraint verification as well as in [2] for KG-specific SPARQL generation. Many of those works emphasize the need of deeper investigation and more systematic test scenarios. Therefore, the development of new benchmarking frameworks and generators currently is a topic of high relevance. The Open LLM Leaderboard [1], aims to evaluate the impressive performance claims of LLMs using the 7 Key Benchmarks from the *Eleuther AI Language Model Evaluation Harness*[17]. A highly promising approach within a sub-discipline of LLM-driven KGC is highlighted in Text2KGBench [14] and measured using a benchmark developed for that purpose.

A hardly-researched aspect of benchmarking LLMs is the evolution and shift of model performance over time [21]. LLM services have been observed to substantially change within a relatively short amount of time, forcing users to continuously adapt prompts, settings or even model choices as to keep the performance to their respective downstream applications stable. This is especially challeng-

ing because, it is currently not transparent how exactly LLMs like GPT-3.5 or GPT-4 are updated and how model behavior will be affected for its different, multi-faceted capabilities [3]. In consequence, there is the need for empiric studies, which try to systematically assess skills of LLMs for different perspectives of KGE. At the time being, though, the task of systematically evaluating model performance states itself a real challenge, as standard approaches and procedure, like regression testing, need to be substantially re-examined in the context of LLMs, e.g. due to different correctness notions, prompting brittleness, and non-determinism [9].

## 3 Assessment Setup

We performed 6 tasks via LLM-KG-Bench 1.2 [11], that assess individual skills of LLMs to read, understand, analyze, and create KGs using Turtle or SPARQL. The tasks are executed in two different manners, T1, T2, T5 and T6 are executed as *static* tasks (fixed problem size) 20 times per model, while T3 and T4 are *scalable* in problem size and are executed 20 times per combination of size and model for 8 different sizes.

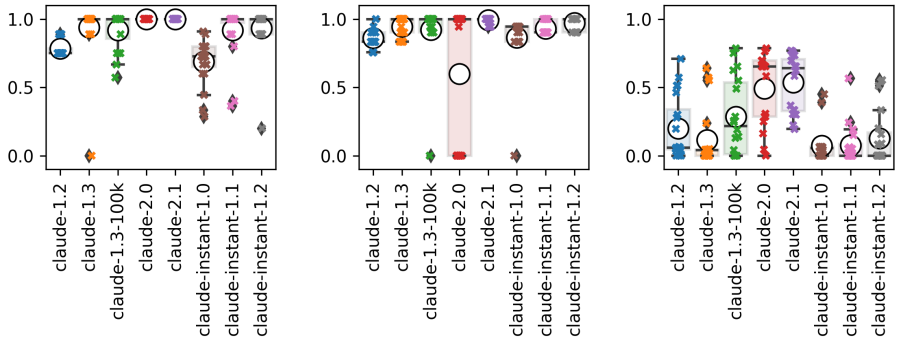
Additionally, for the original tasks from [4] we performed a replication study using the same model versions. Due to the randomness with the default temperature, there is a slight variation but the results remain in the same interval (see [5]).

**Task T1 - Find Connection in a Turtle Org Graph** checks basic understanding for the RDF-KG data model and Turtle reading skills, and asks to find the shortest connection from *:Anne* to *:Bob* in a small organizational graph. For reasons of brevity, we refer the reader to the task fact sheets[3] or [4] for the input documents and prompts. F1 score is measured for the list of IRIs mentioned in the model response with regard to the list of IRIs representing the nodes of the shortest path.

**Task T2 - Find Syntax Errors in Turtle Org Graph** is based on the same file from T1 but has a period missing and a semicolon in another line was removed. We prompt to correct the error without altering the formatting. Correcting the errors demonstrates the LLM's knowledge of Turtle grammar while also showing its ability to transform it into a proper form without altering existing facts and adhering strictly to the task requirements like answering with just the corrected turtle document and keeping the original formatting. F1 is calculated for the parsable, normalized triples, comparing the LLM's answer with the perfect answer.

**Task T3 - Create Example Person Graph** requests to generate a KG using the `FOAF` vocabulary with $n$ (ranging from 10 to 80 with step size 10) persons who each have between 2 and 5 friends . The task is motivated by the idea of using LLMs to generate test, training, or example data of various sizes for KGE steps, but also assesses Turtle writing and RDF-KG modeling skills

[3] https://github.com/AKSW/LLM-KG-Bench/tree/main/LlmKgBench/bench/




(a) T1 - Connected Path (b) T2 - Turtle Fixing (c) T5 - Fact Extraction

Fig. 1: Claude Evolution for Static Tasks: Distribution of F1 scores

(rdf:type, etc.). Although multiple scores were evaluated, we report a relaxed normalized score `person_relative_error`, which is 0 iff number of `foaf:Person` matches $n$, $> 0$ if there are more, $< 0$ if there are less persons than $n$.

**Task T4 - Count Friends in Person Graph** asks to name the IRI of the person with most incoming foaf:knows edges, given a simple KG with $n$ persons (6, 16,..., 76). Each person is known by two other persons, but one designated `foaf:Person` is known by three additional persons (one for size 6), resulting in 5 (resp. 3 for size 6) incoming links instead of 2. This task tests for graph comprehension (direction of edges) and processing skills by aggregating link counts for various KG sizes. F1 is reported w.r.t. the expected person IRI.

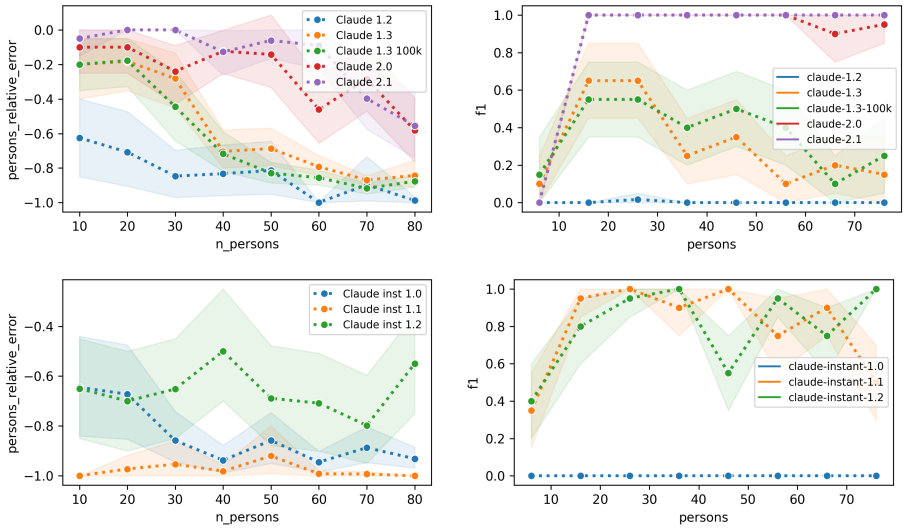
**Task T5 - Create KG from Factsheet** assesses the LLM's fact extraction and advanced RDF modeling abilities, by requesting to generate a Turtle file that captures a subset of information from a 3D printer spec PDF plaintext excerpt.

The prompt is designed to be very specific and unambiguous on how the data should be represented, but also challenges knowledge about ontologies. We evaluate F1 measure, comparing the set of parsable normalized triples to the reference document.

**Task T6 - Wikidata SPARQL query generation** tests SPARQL syntax formulation skills as basic strategy to extract information from a (larger) KG. Leveraging LC-Quad [19], we provide a natural text input query, along with a mapping for all IRIs, that occur in the reference SPARQL query, to their English labels. We calculate F1 by comparing the result values with values of the reference query.

## 4 Claude Models Evolution

For **T1**, Fig. 1a shows that Claude 2.0 and 2.1 reply consistently with accurate answers. All Claude 1.x versions give a few incorrect answers. Claude 1.3 performs better than Claude 1.2 but both miss expected nodes in the answer. This




(a) T3 - Person Graph Creation

(b) T4 - Friends Counts

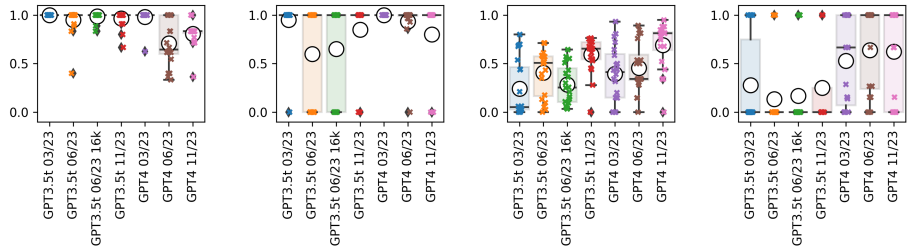
Fig. 2: Claude Evolution for Scalable Tasks: Mean of task metric - 95% CI

effect can also be observed for the Instant versions, but additionally, nodes are reported twice in the answer. There is a light trend of performance improvement for every model evolution.

As can be seen in Fig. 1b for **T2**, Claude models improved over time within the three model lines Claude Instant 1.x, Claude 1.x and Claude 2.x. However, while Claude 2.1 performs best (although sometimes syntax errors remain), Claude 2.0 performs worst. Several ratings of 0 for F1 were caused by too much modifications to the original formatting and structure.

Claude 1.2, as well as all Instant versions struggle with **T3** for all graph sizes (see Fig. 2a). However, for the 2.x versions, the performance is significantly more stable with increased graph size. Claude 2.1 delivers the best answers and the correct amount of persons for sizes 20 and 30. The larger context size of Claude 1.3 100k did not help improve the quality of the results. There is a clear tendency for improvement for all non-Instant Claude version iterations. However, Claude inst. 1.1 performed slightly worse than its predecessor.

Considering **T4**, all versions seem to be challenged by the size of 6 persons as indicated in Fig. 2b. In this tricky case, only 2 persons (instead of 4) are different compared to the other persons. The models often respond with the incorrect person having the most outgoing instead of ingoing edges. Claude 1.2 is struggling over all sizes and in all iterations to name the correct Person. Similarly, it proposes one of the 3 persons that have one more outgoing link to the designated target person (that has the most friends) in almost every iteration. Both Claude 1.3 versions also have this confusion but only in around half of the




(a) T1 - Path (b) T2 - Turtle Fix (c) T5 - Extraction (d) T6 - SPARQL

Fig. 3: GPT-3.5 and GPT-4 evolution for Static Tasks: Distribution of F1 scores

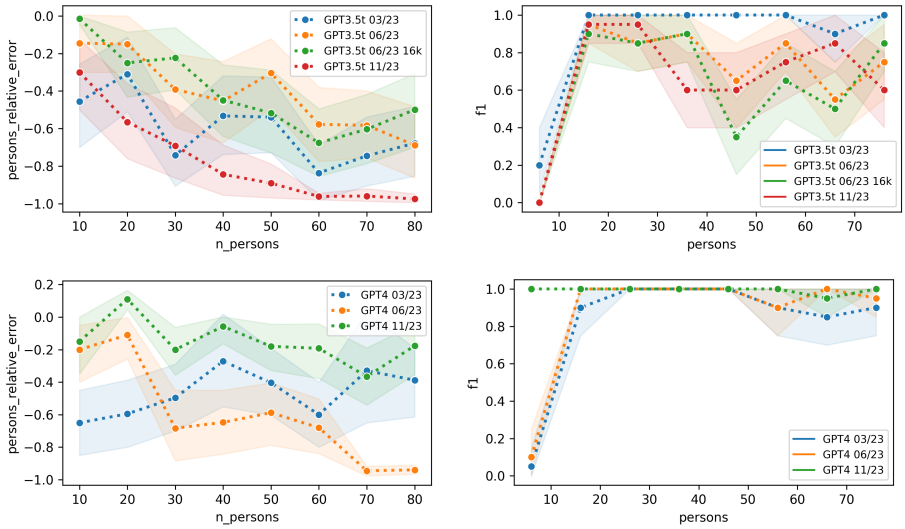
cases, with a slight tendency of doing this error more often with increased person size. For Claude 2.1, this confusion does not happen for all larger sizes and for Claude 2.0 only in 2 instances. As shown in Fig. 2b(lower), Claude Instant 1.0 performs similar to Claude 1.2. Claude Instant 1.2 gives a slight performance improvement, by being more reliable and consistent and having less confusion. In general, we observed in Task 4, that every new version brought an improvement. When comparing the Instant versions to the full Claude version it is noteworthy that the Instant versions seem to adhere more strictly to the output format request.

For **T5**, Fig. 1c shows that Claude 1.2 is on median and average slightly better than its direct successor (which produces invalid Turtle in almost half of the cases that leads to zero triples using our recovery parsing heuristic), but worse than Claude 1.3-100k. All Instant versions struggle to produce valid Turtle, leading to very few Triples that can be extracted (approx. half of all the instances lead to zero recovered triples). There is a slight improvement between every Instant version: if triples can be recovered, the number of triples extracted increases with every version step. The Claude 2.x versions do not make severe syntactical errors that render entire documents as unusable. Unfortunately, the trend that was discovered in [4], that Claude 2.0 severely violates the output constraints by giving explanations or titles in the first lines ("here is your RDF:") also materialized in version 2.1.

The Claude LLMs seem to have severe problems with **T6**. While most of the time syntactically valid SPARQL is produced, the execution just returns empty result sets. This is caused by semantic errors. Most often we have seen Claude mixing up Wikidata IRIs or using them in a wrong fashion. There has been only one instance with a correct result. In favor of readability, we refrain from showing a plot for this task.

## 5 GPT-3.5 and GPT-4 Models Evolution

As indicated by Fig. 3a, GPT versions from March 2023 seem to tackle **T1** very well (only one mistake by GPT-4). This shows that the effect of a decreased




(a) T3 - Person Graph Creation

(b) T4 - Friends Counts

Fig. 4: GPT-3.5/4 Evolution for Scalable Tasks: Mean of task metric - 95% CI

performance for newer model versions, which has been reported in several online posting as well as in [9], also affects KGE workloads. For later versions we experienced problems like additional or missing list entries. There is a very clear performance drop for GPT-4 6/23, which unfortunately has not been resolved entirely for the November version.

**T2** reveals a similar pattern, that the performance for the March versions is the best (see Fig. 3b). We discovered two main error classes: performing no fix but stating no error would exist, or changing the turtle formatting too much. For GPT 3.5 typically the first class could be observed. This seems to happen especially with the 6/23 versions from GPT 3.5 (4k and 16k). In contrast, GPT 4 seems to ignore the instruction to keep the original layout. In certain instances, it modifies the turtle format so drastically that the evaluation code fails to retrieve the turtle snippet from the response, resulting in a penalty of 0 values for F1.

As shown in Fig. 4a for **T3**, except for the March releases, all models perform relatively well for the smallest graph size and we see a decrease in answer quality as the size of the output increases. This trend is not significant for the March versions. Interestingly, the latest version of GPT-3.5 demonstrates the poorest performance among all models, opposed to the latest GPT-4 version showing the best performance. While GPT-4 6/23 exhibits a decline in performance relative to its predecessor like in T1 and T2, the situation is reversed for GPT-3.5.

For **T4** we found w.r.t. GPT-3.5 (see 4b, upper), that version 3/23 is the only one that in some iterations is able to name the correct person in the tricky case of the smallest size, but it also very clearly outperforms all newer versions for the

remaining sizes. Only for the size of 66 persons there are two erroneous responses where it chooses the first person. Picking Person-1 (which has no special characteristic other than its position at the beginning of the file) seems to be the most common error for all newer GPT-3.5 versions. While there is a tendency that the 16k version performs worse compared to the default (4k) version of 6/23 for the tested sizes, there is no clear ranking between the remaining versions, given the confidence interval and interleaving mean curves. In contrast, our observations of the GPT-4 versions presented a reverse scenario (see Fig. 4b, lower). The novel November version reports only one incorrect person in the entire test, showing an almost perfect performance. It is the only version tested, that dealt correctly with the tricky case. Version 6/23 made an error, in 3 instances (Person 1 as answer), while the majority of incorrect answers of the March version (that performs worse) are due to the incoming vs. outgoing edges confusion.

In **T5** we can see a clear improvement trend from old versions to new versions for both GPT-3.5 and GPT-4 versions (see Fig. 3c). The version of 6/23 that supports a 16k context window, has a slightly worse performance than the version with default context size (4k). The pattern observed for Claude, that it does not respond with the plain turtle, was also experienced for the GPT versions. While the GPT-3.5 versions correctly reply with the plain turtle, the GPT-4 versions sometimes wrap the answer in Markdown code blocks. However, starting for the November version of GPT-4 this is consistent for all runs and in a unified format.

**T6** seems to be quite difficult for GPT-3.5 and GPT-4 as can be seen in Fig. 3d. SPARQL queries produced by all GPT-3.5 versions often produce no result, but roughly every fourth answer is perfect. The GPT-4 versions have a better probability and create partial correct results as well.

## 6 Conclusion and Future Work

From the perspective of basic skills for KGE, the latest cutting-edge versions (GPT-4T 11/23, Claude 2.1) demonstrated for the majority of tasks improved capabilities over all their predecessors. However, we provided evidence that the phenomenon which has been reported in blogs and literature, that for specific tasks the performance degraded over the course of 2023 also applies for selected KGE workloads (especially for the GPT March/June evolution). One troubling reason is, that the newer models, although explicitly requested, do not consistently respond in the specified output format (e.g. plain Turtle) but include short explanations or markdown ticks. This impedes interfacing these models from code and requires complex and failure tolerant extraction and parsing routines. Since we observed very different behavior, this additionally hinders a plug-and-play integration of various LLM (versions) into tools. Therefore, we see as a next step to define multi-shot tests that are stricter, however provide feedback to the models (e.g. parsing errors). Moreover, it could be of interest to assess the performance using N-Triples (allowing easier data extraction from partially inconsistent responses), or JSON-LD (popular for websites employing schema.org) and adding other KGE task (e.g. RML, SHACL, more SPARQL tasks).

**Acknowledgments.** This work was partially supported by grants from the German Federal Ministry for Economic Affairs and Climate Action (BMWK) to the KISS project (01MK22001A) and CoyPu project (01MK21007A) as well as from the German Federal Ministry of Education and Research (BMBF) to the project StahlDigital (13XP5116B).

## References

1. Beeching, E., Fourrier, C., Habib, N., Han, S., Lambert, N., Rajani, N., Sanseviero, O., Tunstall, L., Wolf, T.: Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard (2023)
2. Brei, F., Frey, J., Meyer, L.P.: Leveraging small language models for text2sparql tasks to improve the resilience of ai assistance. In: D2R2'24: Third International Workshop on Linked Data-driven Resilience Research 2024 @ ESWC2024 (2024)
3. Chen, L., Zaharia, M., Zou, J.: How is ChatGPT's behavior changing over time? (Oct 2023), http://arxiv.org/abs/2307.09009, arXiv:2307.09009 [cs]
4. Frey, J., Meyer, L., Arndt, N., Brei, F., Bulert, K.: Benchmarking the abilities of large language models for RDF knowledge graph creation and comprehension: How well do llms speak turtle? In: Alam, M., Cochez, M. (eds.) Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2023) co-located with the 21th International Semantic Web Conference (ISWC 2023), Athens, November 6-10, 2023. CEUR Workshop Proceedings, vol. 3559. CEUR-WS.org (2023), https://ceur-ws.org/Vol-3559/paper-3.pdf
5. Frey, J., Meyer, L.P., Brei, F., Gründer-Fahrer, S., Martin, M.: LLM-KG-Bench-Results: 2024 ESWC LLM Evolution 2023 Results. https://github.com/AKSW/LLM-KG-Bench-Results/blob/main/2024-ESWC_LLM-Evo-2023/Readme.md (2024)
6. Frey, J., Meyer, L.P., Brei, F., Gründer-Fahrer, S., Martin, M.: LLM-KG skills evolution time capsule 2023 for Claude & ChatGPT - Results and log of LLM-KG-Bench runs (Jan 2024). https://doi.org/10.5281/zenodo.10572907
7. Hofer, M., Frey, J., Rahm, E.: Towards self-configuring knowledge graph construction pipelines using llms-a case study with rml. In: Fifth International Workshop on Knowledge Graph Construction @ ESWC2024 (2024)
8. Hu, X., Tian, Y., Nagato, K., Nakao, M., Liu, A.: Opportunities and challenges of chatgpt for design knowledge management (Apr 2023). https://doi.org/10.48550/ARXIV.2304.02796
9. Ma, W., Yang, C., Kästner, C.: (why) is my prompt getting worse? rethinking regression testing for evolving llm apis (2023)
10. Martin, A.: Challenges requiring the combination of machine learning and knowledge engineering (2023), https://ceur-ws.org/Vol-3433/preface.pdf
11. Meyer, L.P., Frey, J., Arndt, N., Junghanns, K., Brei, F., Stadler, C.: Aksw/llm-kg-bench: 1.2.0 (Dec 2023). https://doi.org/10.5281/zenodo.10302307
12. Meyer, L., Frey, J., Junghanns, K., Brei, F., Bulert, K., Gründer-Fahrer, S., Martin, M.: Developing a scalable benchmark for assessing large language models in knowledge graph engineering. In: Keshan, N., Neumaier, S., Gentile, A.L., Vahdati, S. (eds.) Proceedings of the Posters and Demo Track of the 19th International Conference on Semantic Systems co-located with 19th International Conference on Semantic Systems (SEMANTiCS 2023), Leipzing, Germany, September 20 to 22, 2023. CEUR Workshop Proceedings, vol. 3526. CEUR-WS.org (2023), https://ceur-ws.org/Vol-3526/paper-04.pdf

13. Meyer, L.P., Stadler, C., Frey, J., Radtke, N., Junghanns, K., Meissner, R., Dziwis, G., Bulert, K., Martin, M.: Llm-assisted knowledge graph engineering: Experiments with chatgpt (2023), to appear in conference proceedings of AI-Tomorrow-23, 29.+30. 6.2023 in Leipzig, Germany
14. Mihindukulasooriya, N., Tiwari, S., Enguix, C.F., Lata, K.: Text2KGBench: A Benchmark for Ontology-Driven Knowledge Graph Generation from Text (Aug 2023). https://doi.org/10.48550/arXiv.2308.02357, arXiv:2308.02357 [cs]
15. Pan, J.Z., Razniewski, S., Kalo, J.C., Singhania, S., Chen, J., Dietze, S., Jabeen, H., Omeliyanenko, J., Zhang, W., Lissandrini, M., Biswas, R., de Melo, G., Bonifati, A., Vakaj, E., Dragoni, M., Graux, D.: Large Language Models and Knowledge Graphs: Opportunities and Challenges (Aug 2023), http://arxiv.org/abs/2308.06374, arXiv:2308.06374 [cs]
16. Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., Wu, X.: Unifying large language models and knowledge graphs: A roadmap (Jun 2023)
17. Sutawika, L., Schoelkopf, H., Gao, L., Biderman, S., Abbasi, B., Tow, J., ben fattori, Lovering, C., farzanehnakhaee70, Phang, J., Thite, A., Fazz, Aflah, Muennighoff, N., Wang, T., sdtblck, gakada, nopperl, researcher2, tttyuntian, Chris, Etxaniz, J., Kasner, Z., Khalid, Hsu, J., Lee, H.A., Kanekar, A., AndyZwei, Ammanamanchi, P.S., Groeneveld, D.: Eleutherai/lm-evaluation-harness: v0.4.1 (Jan 2024). https://doi.org/10.5281/zenodo.10600400
18. Trajanoska, M., Stojanov, R., Trajanov, D.: Enhancing Knowledge Graph Construction Using Large Language Models (May 2023), http://arxiv.org/abs/2305.04676, arXiv:2305.04676 [cs]
19. Trivedi, P., Maheshwari, G., Dubey, M., Lehmann, J.: Lc-quad: A corpus for complex question answering over knowledge graphs. In: International Semantic Web Conference. pp. 210–218. Springer (2017)
20. Zhang, B., Reklos, I., Jain, N., Peñuela, A.M., Simperl, E.: Using large language models for knowledge engineering (llmke): A case study on wikidata (2023)
21. Zheng, S., Zhang, Y., Zhu, Y., Xi, C., Gao, P., Zhou, X., Chang, K.C.C.: Gpt-fathom: Benchmarking large language models to decipher the evolutionary path towards gpt-4 and beyond (2023)
22. Zhu, Y., Wang, X., Chen, J., Qiao, S., Ou, Y., Yao, Y., Deng, S., Chen, H., Zhang, N.: Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities (2023)