LLMs for Knowledge-Graphs enhanced Task-Oriented Dialogue Systems: Challenges and Opportunities

Vasile Ionut Remus Iga[1][0009−0001−4568−929X] and
Gheorghe Cosmin Silaghi[1][0000−0002−3447−4736]

Babeş-Bolyai University
Business Informatics Research Center, Cluj-Napoca, Romania
vasile.iga@ubbcluj.ro, gheorghe.silaghi@ubbcluj.ro

Abstract. Large Language Models are a great tool for solving diverse tasks formulated in natural language. Recent work has demonstrated their capacity of solving tasks related to Knowledge Graphs, such as Knowledge Graph Completion or Knowledge Graph Reasoning, even in Zero- or Few-Shot paradigms. However, given a particular input, they do not always produce the same output, and sometimes point to intermediate reasoning steps that are not valid, even if they produce a satisfactorily answer. Moreover, the use of LLMs is mostly studied for static knowledge graphs, while temporal ones are overlooked. To highlight opportunities and challenges in knowledge graph related tasks, we experiment with ChatGPT on graph completion and reasoning for both static and temporal facets, using three different prompting techniques in Zero- and One-Shot contexts, on a Task-Oriented Dialogue system use case. Our results show that ChatGPT can solve given tasks, but mostly in a non-deterministic way.

Keywords: Large Language Models · Knowledge Graph · Knowledge Graph Completion · Knowledge Graph Reasoning · Task-Oriented Dialogue System.

1 Introduction

Knowledge Graphs (KG) can be defined as graphs of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and edges represent potentially different relations between these entities [3]. Two main categories of KGs are of our interest: static and temporal. The former captures the information valid at a certain moment in time, similar to a snapshot, while the latter is more dynamic, adding the temporal context to each fact [6] i.e. duration for the knowledge validity. To keep the KG consistent and valid and to perform inference on the stored knowledge base, two tasks are especially important: Knowledge Graph Completion (KGC) and Knowledge Graph Reasoning (KGR). KGC aims to infer missing facts in a given KG [9].

It can done be either from input text or from already existing knowledge. KGR focuses on generating an answer that is grounded with facts stored in KG.

In the last years, chatbots and task-oriented dialogue systems (TODs) gained an incredible popularity, all culminating with the well-known ChatGPT[1], as the latest expression of the Artificial Intelligence advances. While chatbots are mostly used for simple chatting without a specific goal in mind, TODs aim to solve the user's specific tasks within certain domains

In our previous work [5] we developed an ontology-enhanced TOD that uses a static KG to map the context of the discussion and store relevant information. This innovation leads to important advantages, such as the possibility of having multiple threads of discussion in the same conversation, and the KG acting as a proxy for data validation. Eventually, the system can help the company to construct and manage its specific knowledge base and perform several tasks on it, such as the Create-Retrieve-Update-Delete (CRUD) operations. With the help of our TOD system, we are able to perform the tasks of KGC to construct the KG, and KGR to solve the CRUD operations.

However, our system based its abilities on text template-matching rules, which limited the naturalness of dialogues, and the possibility of adapting to new concepts that are outside the provided ontology. Therefore, in a subsequent work [4], we trained neural networks to detect the intent of the user and relevant related entities from the input text. Although it has shown promising results, this approach didn't fully solve the aforementioned shortcomings. Another limitation is the use of static KGs, which cannot capture time-related validity of facts, as opposed to temporal KGs.

Therefore, in our current work, we study the usage of Large Language Models (LLMs) to solve the KGC and KGR tasks, in the context of the our TOD system. KGs and LLMs have shown possible integration, as KGs can enhance LLMs by providing external knowledge for inference and interpretability, while LLMs can solve KG-related tasks using natural language prompts [9].

Our experiments explore LLMs for KGC and KGR in both static and temporal contexts. Through the use of ChatGPT, we test its capabilities of solving the mentioned tasks using three different prompting techniques, in two contexts: Zero-Shot (ZS) Direct Prompting (DP) and Chain of Thought (COT), and One-Shot (OS) In-Context Learning (ICL) and Chain of Thought. To highlight a proper use case for LLMs to KG tasks, example phrases are extracted from the training phase of the TOD System. In this way, we not only test whether the LLM can solve KG-specific tasks, but we also explore its integration with our dialogue system.

The initial experiments run in the current paper aim to test the water about the interconnection between the mentioned technologies, thus paving the way for further, more robust experiments. Therefore, as stated in [10,1], positive feedback from initial testing may show that LLMs can be used as a new type of interface for humans to interact with any kind of systems through natural language. LLMs

[1] `https://openai.com/blog/chatgpt`

may enhance a TOD system that helps any subject-matter experts in fulfilling their roles.

Our research brings in the following contributions: (i) We evaluate ChatGPT for the KGC and KGR tasks, in both static and temporal KGs, using three different prompting techniques (DP, ICL, COT) in two data contexts (Zero- and One-Shot), revealing insights on how a powerful LLM perform such tasks. (ii) We test whether such models can be integrated in a domain-specific ontology-enhanced TOD system, by extracting and using test phrases specific to its training.

The paper evolves as following: in section 2 we describe recent works about solving the KGC and KGR tasks using general and LLM-based approaches, on both static and temporal KGs. Next, section 3 presents our methodology, describing each steps of our experiments, the used examples, ontology, and prompts. Section 4 discusses the obtained results, while section 5 concludes the paper.

2 Related Work

Ji et al. [6] present solutions for the KGC task on static KGs using embedding-based models (ex. TransE), relation path reasoning (ex. Path-Ranking Algorithm), reinforcement-learning path finding (ex. path-finding between entity pairs as a Markov Decision Process), rule-based reasoning (ex. KALE), or meta relational learning (ex. using R-GCN or LSTM). For KGR, neural network approaches are presented, for single and multi-hop question answering. Similar findings are presented by Zhang et al. [12] while classifying them as neural, symbolic and neural-symbolic.

Temporal KGs require more sophisticated systems, therefore new or refinement ones were further engineered. Wang et al. [10] present solutions for temporal KGC, including interpolation methods (i.e. estimating missing values from known ones) such as timestamps dependent-based, timestamps specific functions-based, or deep learning-based, and extrapolation methods (i.e. predicting future facts based on known ones) such as rule-based, graph neural network-based, meta-learning-based, or reinforcement learning based. Liang et al. [7] distinguish between RNN-based (using RNNs, LSTMs or GRU networks) or RNN-agnostic models (time-vector guided or time-operation guid-ed) for Temporal KG.

All of the previous presented works emphasize the use of neural networks, logic networks, logic rules or mathematical operations to solve KGC and KGR in static and temporal KGs, but none of them actually focus on using LLMs. Pan et al. [9] studies the synergy between LLMs and KGs, proposing a unified roadmap, where KGC and KGR are also tackled. Zhu et al. [14] experimented with ChatGPT and GPT4 for KGC and KGR, concluding that they are below state-of-the-art (SOTA) fine-tuned Pre-Trained Language Models (PLMs) for KGC in a zero/one shot paradigm, but their reasoning capabilities are either close or above SOTA models. Still, the usage of an LLM and its efficiency compared to a specialized PLM is unclear. Han et al. [2] propose PiVE, a prompting technique, where an LLM (ChatGPT) extracts facts from input texts while a smaller fine-tuned PLM checks and completes its responses in an iterative man-

ner. In a somewhat similar fashion, Wei et al. [11] proposes a multi-stage dialogue with ChatGPT to extract relevant information from input texts, given a certain schema.

Similar to Wei et al.[11], we test the capacity of a LLMs (ChatGPT) on KGC and KGR tasks. We depart from them as we extend the analysis from static to temporal KGs. Moreover, we also test the possibility of integrating the LLM with an ontology-enhanced TOD system, to sharpen its natural language processing and KG-related capabilities, by using example phrases from its training schedule.

3 Methodology

In this section, we describe how we assess that ChatGPT is fit for the study at hand using preliminary questions, the ontology used to anchor's the LLMs knowledge, setup format for the static/temporal KGC and KGR tasks, and prompting techniques with examples. The ontology and extracted triples are described in RDF, using the Turtle syntax. ChatGPT 3.5 is used, while prompts are manually designed. Each individual prompt with each example is fed three times to ChatGPT in the same conversation, for three different sessions. In this way, we test the inter- and intra-conversation determinism of the LLM. All prompts are available at `https://github.com/IonutIga/ChatGPT-KGC-KGR/tree/main`.

3.1 Preliminary questions for ChatGPT

Before experimenting with ChatGPT, we decided to question its knowledge to assess if it is fit to solve our intended KG-related tasks. First, we design six questions related to the tasks at hand, such as:

1. Do you know what a Knowledge Graph is?
2. Do you know what a Temporal Knowledge Graph is?
3. Do you know what an ontology is, in the context of Knowledge Graphs?
4. Do you know what Turtle syntax is, in the context of Knowledge Graphs?
5. Do you know what a triple is, in the context of Knowledge Graphs?
6. Do you know how to extract triples from a natural language phrase, given a provided ontology in the Turtle syntax?

ChatGPT responded in an organized, clear, and factual manner to each of the above questions, demonstrating its understanding of KGs, Temporal KGs, Turtle syntax, ontologies and tasks at hand. Two additional questions were asked, related to time:

7. Do you know today's date?
8. Do you know today's time?

Today's date was not a problem to ChatGPT, also being able to respond to subsequent questions regarding yesterday or tomorrow's date, given a certain one. However, it couldn't respond to time-related questions, stating "*I don't have real-time capabilities, including the ability to provide the current time in specific time zones*".

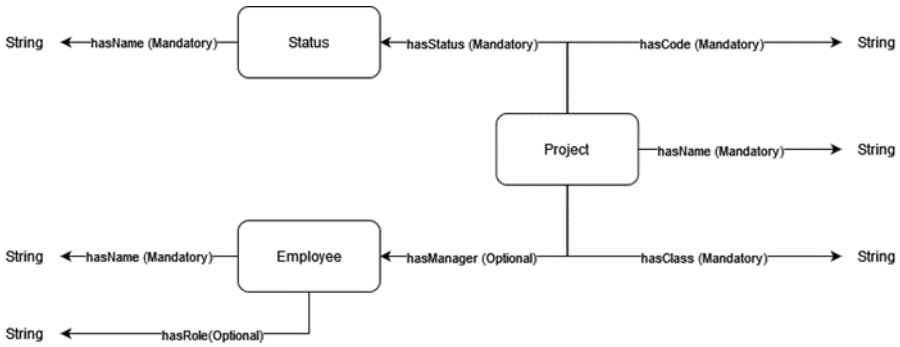

Fig. 1. The ontology used throughout the experiments. It contains three classes and six relationships

3.2 Ontology

We use the ontology from our previous work [5] (see Fig. 1). It contains three classes (i.e. concepts): *Project*, *Employee*, and *Status*, and six relationships between them (*hasManager, hasStatus*) or classes and literal values (*hasName, hasRole, has-Class, hasCode*).

3.3 Setup format

The ontology is written using the Turtle syntax, where each fact has the form of a triple (subject, relationship, object), therefore each of the extracted facts are aligned to the same format.

For the Temporal KGs, inspired by the work of Nguyen et al. [8], each relationship becomes an instance of its actual type, while two new temporal relationships are added to it: *startDate* and *endDate*. The start date is either today (when the input text is encountered) or a provided date. The end date is either "*Now*" - highlighting the continuous knowledge validity, or a provided date. For example, suppose we have the triple `:Employee1 :hasName :John`. The relationship `:hasName` will be instantiated as `:hasName1 rdf:instanceOf :hasName.` Then, the two temporal relationships are defined, as `:hasName1 :startDate "20-02-2024"`, `:hasName1 :endDate "Now"`, meaning the fact where `:hasName1` is valid from 20-02-2024 until further notice.

For KGR, ChatGPT has to use the provided KG to answer the input question, ensuring its time validity. For example, given the above example, a question such as "*Is there an employee named John*?" requires ChatGPT to find any employee named "*John*" and compare its start date and end date with today's date.

3.4 Prompting techniques and examples

Three prompting techniques are used to communicate with ChatGPT, namely Direct Prompting, In-Context Learning, and Chain of Thought. All three follow the guidelines of Zhao et al. [13], where a prompt must be detailed enough,

```
You are an expert in Knowledge Graphs. Turtle is used as the syntax language. An ontology is
provided to you. It contains classes and relationships between classes or classes and literals (strings,
numbers, dates etc.). A natural language phrase is the input to you.
It may or may not contain references to an instance of a class provided in the ontology, together
with specific relationships.
Your task is to extract information from the input phrase about the instance of a class using the
provided ontology, in the form of a triple (subject,relationship, object), using the Turtle syntax.
Each instance should be identified by an ID, using the format Class + DDMMYYYY + R, where
class is the detected class, + is concatenation, DDMMYYYY is the format for today's date, and
R is a random number between 0-100.
The first triple you always create is between the ID and the class type, using the rdf:type relation-
ship. Next, each extracted information that is related to the detected instance should use the ID
as the subject.
Make sure each triple is in line with the ontology, meaning each class type and relationship that
is detected should be present in the ontology.
The ontology:
The input phrase:
```

Fig. 2. The standard prompt template used among all techniques

but straightforward, with no negative directives (i.e. only telling what to do), adding relevant examples to the task and input phrase when needed. Direct Programming assumes no further information than the task description, context information, ontology, and input phrase are fed to ChatGPT. In-Context Learning adds one relevant example to the prompt, while Chain-Of-Thought requires the model to think step by step about the solution to the task at hand, theoretically enhancing its focus on it. As the aim of our study is Zero- and One-Shot paradigms, we obtain four possible combinations:

- Zero-Shot Direct Prompting
- Zero-Shot Chain of Thought, by only adding "*Let's think step by step*" to the prompt
- One-Shot In-Context Learning, adding one relevant example to the prompt
- One-Shot Chain of Thought, adding one relevant example, together with reasoning steps to the prompt.

Each technique uses a standard prompt, depicted in Fig. 2. It starts by saying *"You are an expert in Knowledge Graphs"*, that should guide ChatGPT to use adequate knowledge. Next, we supply ChatGPT with details about the ontology and the format, together with the task description. In our previous work [5], each instance's ID in the KG had a similar format to "Class name + DDMMYYY + R", where we concatenate the name of instance's type, today's date - when the input phrase is encountered using the DDMMYYYY format, and R i.e. a random number between 0-100. Lastly, some verification guidelines are provided.

For KGC on temporal KGs, additional text is added to guide ChatGPT in constructing proper temporal-enhanced triples, as presented in subsection 3.3. First, it becomes *"an expert in Temporal Knowledge Graphs"*. Then, we guide ChatGPT how to construct each relationship instance's ID, by concatenating the name of the relationship type with the random number between 0-100, similarly to class instances' IDs.

Table 1. Input phrase examples and their types

Input phrase example	Input phrase type
Insert an employee with name as Michael and role as CEO.	explicit Information
It seems I cannot find Michael as an employee, therefore you should add him. Remember, he's our CEO.	implicit Information
Michael is not on the list of empyees, but it should, as it is our CEO.	misleading information
Michael is a very good boy, doing a good job as a guide dog.	misleading information
Assuming today's date is 01-01-2010, what is Michael's role?	before validity time
Assuming today's date is 01-01-2030, what is Michael's role?	after validity time

For KGR, the prompt is straightforward: *"You are an expert in Temporal Knowledge Graphs. Your task is to answer the input question based on the provided knowledge graph. Knowledge Graph: ... Input question:".*

Table 1 presents each input phrase used throughout experimenting. As mentioned before, all the input phrases are extracted from the training schedule of our TOD System. Three input phrases types are tested: explicit in-formation, implicit information, and misleading information. Each aims to reveal ChatGPT's capacity of dealing with explicit phrases where the instances and relationships are easy to spot, implicit ones that need some additional reasoning, and misleading examples where either mistakes or out-of-ontology class types are used.

For temporal KGs, we use the same input phrase example as the one for explicit information. For KGR, we asked ChatGPT two additional questions, to test its reasoning capabilities on Temporal KGs, where we assumed the date of questioning it is outside or inside the validity time interval specified in the provided KG.

4 Results and discussion

In this section, we discuss the overall conclusions extracted from our experiments. Each of them are grouped based on the task at hand.

4.1 ChatGPT on the static KGC task

We analyze the results of ChatGPT on the Static KGC task, by breaking down to each input phrase type and technique.

Explicit Information

Regarding explicit information phrase type with Zero-Shot Direct Prompting, ChatGPT was able to extract the target triples from the input phrase, and was consistent on output and explanations through each conversation. However, it had difficulties in generating a random number R and concatenate it to the instance's ID.

On Zero-Shot Chain of Thought, ChatGPT had no problems extracting the triples, providing detailed explanations of the reasoning process. Results were consistent intra- and inter-conversations, having the same problem with the random number R concatenation.

The model followed the provided example when prompted with One-Shot In-Context Learning technique, having as output only the triples (one conversation added short explanations for each fact). The ID is correctly formulated, while results were consistent throughout conversations.

One-Shot Chain of Thought paradigm provided the most complete answers, in terms of explanations and output triples. As expected, ChatGPT followed the detailed step-by-step reasoning process given in the prompt. Results were consistent inter- and intra-conversations.

Overall, the explicit information type phrase was easy to work with for ChatGPT, mostly following the provided guidelines. Some problems regarding the number R concatenation were faced with ZS DP and COT, which is surprising, as it was the simple tasks of all. Consistency between chats and conversations was mostly kept, making ChatGPT reliable for such task.

Implicit Information

With Zero-Shot Direct Prompting, ChatGPT faced the first problems in outputting a correct set of triples. As a result, in two out of three conversations, it added one more fact stating that the instance is also of type owl:Class, giving a wrong explanation. Thus, is not consistent inter-conversation, only in the same dialogue session. It has the same aforementioned problems with the number R concatenation.

Zero-Shot Chain of Thought raised some interesting problems. First, ChatGPT missed the name fact two out of three conversations. Next, it didn't generate neither correct date or R in any conversation. Therefore, the results were not consistent at all.

One-Shot In-Context Learning fixed the issues from above, outputting correct sets of triples, following the provided examples. However, we assume the model made no fundamental changes in the reasoning process, it just perfectly followed the example. Results were consistent in all conversations.

Finally, One-Shot Chain-of-Thought raised again some of the same problems as in the ZS paradigm. Although conversations were more intra-consistent, between them the name fact was missed, while the date was also not correctly generated throughout one conversation. Finally, explanations were more in-lined with the given prompt.

Overall, the implicit information phrase type has created problems for ChatGPT. Although OS ICL managed to generate proper triples, the other techniques shown the limitations of the reasoning aspect from ChatGPT.

Misleading Information – first example

In practice, some input phrases may contain misspelled words that should still give enough information to generate correct triples. Therefore, we tested ChatGPT with an implicit information phrase type regarding an employee, where the word *"employee"* is misspelled.

ChatGPT shows its limitations from the first set of tests. With Zero-Shot Direct Prompting, the model is able to find some of the right triples, but either

misses the others, or adds statements that are not consistent with the ontology. Moreover, it provides ambiguous explanations regarding the detection of some relationships. Additionally, it doesn't generate the right ID for the instances, by confusing to-day's date and not being able to concatenate the random number.

Surprisingly, Zero-Shot Chain of Thought has similar results with the above technique. Some new problems appear, for example, it suggests that the *"CEO"* role found in the input phrase is just a placeholder for another one. Moreover, results are not consistent throughout conversations. Literature suggests COT should enhance its reasoning process, but it seems it does not work for the task at hand.

In line with other results, One-Shot ICL works best for ChatGPT, as it extracted the right facts, following the provided example. Lastly, One-Shot Chain of Thought alleviates most of the problems from its ZS counterpart. Results become more consistent, while facts are mostly in line with the ontology.

Overall, the main concern is that ChatGPT has encountered more problems by only providing one misspelled word, compared to the implicit information tests. In practice, such use cases would certainly be faced, while ChatGPT fails to dis-play the right behavior.

Misleading Information – second example

To test whether ChatGPT follows the provided ontology and prompts, while also being able to handle Out-Of-Distribution cases, we design an implicit information input phrase where the instance is of type *"Dog"*, that is outside the ontology, and we expect ChatGPT to output such response.

We summarize the results of Zero-Shot Direct Prompting, Zero-Shot Chain of Thought, and One-Shot In-Context Learning, as they are very similar. ChatGPT fails to follow one simple instruction from the prompt, to adhere to the given ontology, and outputs triples that are not factually correct. Some of them borrow relationships from existing classes, while others contain made-up ones by ChatGPT. Explanations are also ambiguous, as it assigns the `hasRole` relationship to an instance because *"it is a guide dog"*. Conversations are not consistent, only by being wrong, while failing to concatenate the R number.

In contrary with the above results, One-Shot Chain of Thought was able to guide ChatGPT into outputting the right explanation (i.e. that no triples were extracted based on the ontology) almost each time. Only one chat of a conversation resulted in wrong triples being extracted.

Overall, ChatGPT was not able to follow basic instructions about dealing with Out-Of-Distribution instances, continued to do some of the same mistakes regarding the generated ID, while being inconsistent in answers. Only OS COT produced relevant results that can be considered correct.

4.2 ChatGPT on the Temporal KGC task

The above section analyzes the results on the static KGC task, while this section takes into consideration the temporal component of Knowledge Graphs. The input phrase is the explicit information type one, as ChatGPT displayed

some issues with more complex ones. The time should be the moment when our examples were run, specifically on February 28, 2024.

Zero-Shot Direct Prompting generated some interesting results. ChatGPT managed to follow the rules of instantiating each relationship and assigning proper time related values, but it was not able to generate correct IDs either for the class or relationship instances. Also, it did not use the `rdf:instanceOf` relationship, as mentioned in the prompt, for the relationship instance fact, but the `rdf:type` one. This mistake is not huge, but it shows that ChatGPT may not follow the guidelines. Moreover, one conversation revealed wrong triples, where a `hasRole` relationship instance was considered as a *Status* class type. Finally, conversations were intra-consistent, but not between sessions.

In accordance with some of the above findings, Zero-Shot Chain of Thought did not improve ChatGPT's output. Moreover, they were inconsistent among conversations, missing triples or adding unnecessary statements. In one occasion, the IDs of relationship instances were far from the provided template. However, time-related relationships were still correctly extracted.

One-Shot In-Context Learning once again highlighted ChatGPT's underlying nature of being a statistical model. The output follows the provided example, by even copying the date in some conversations (although it has explicit guidelines to use the current date). Even the random R numbers assigned to IDs are the same. Although the output is correct, it is doubtful that it is the result of an under-lying reasoning process.

One-Shot Chain of Thought successfully reduced the error from the ZS counterpart, by providing a running example with explicit intermediate reasoning steps. Although this achievement, ChatGPT again copied the current date from the example, without following the prompt on using today's date.

Overall, ChatGPT had no problems with time-related relationships. It successfully anchored each relationship instance in time. However, instantiating each relationship raised problems to the model. The generation of the IDs, together with following the ontology and provided guidelines were the main problems.

4.3 ChatGPT on the KGR task

In this section, we discuss ChatGPT's performance on the KGR task. We analyze two situations, when the time of discussion is assumed to be beyond the knowledge scope of the provided KG, and the other is known.

When the time of discussion is assumed to be beyond the knowledge scope of the provided KG, all techniques managed to lead ChatGPT to valid answers. Each time, all conversations were intra- and inter-consistent, leading to same responses, differing only in formulation. Therefore, ChatGPT could reason correctly about the time validity of certain facts.

When the time of discussion falls in the valid interval of time stated in the KG, ChatGPT faces a few problems with the format of time labeling. We chose *"Now"* to be the value of the `endDate` relationship when the fact is continuously valid to the current date. Because of that, for Zero-Shot Direct Prompting and, surprisingly, for One-Shot In-Context Learning, the model generated responses

where it states that it cannot infer the answer because *"Now"* is not a fixed date and it does not know its meaning. ChatGPT is not entirely wrong, as *"Now"* may be ambiguous, but it is confusing how it could determine it in all other cases, especially during Zero- and One-Shot Chain of Thought.

In conclusion, ChatGPT may be used for temporal reasoning, with additional information for the labeling schema.

5 Conclusion

In line with the findings of Zhu et al. [14], we find that ChatGPT is better at reasoning than extracting facts from input text. The KGR task raised almost no difficulties, as the model was only confused by the definition of the *"Now"* label.

The Static and Temporal KGC tasks brought to surface the limitations of LLMs when it comes to real reasoning. ChatGPT had almost no problems handling explicit type of phrases, but could not follow the provided guidelines and ontology when handling the implicit and temporal ones. Moreover, only ICL was able to generate consistent answers most of the time, but it seems it does not enhance the model's reasoning abilities, only guiding it to follow exactly the provided example. Surprisingly, COT did not help most of the times, which underlines the difficulty of KGC. The temporal aspect did not raise problems to ChatGPT, showcasing its possible use in temporal KGs.

ChatGPT has shown weaknesses when dealing with simple concatenation tasks. Most of the times, it could not generate a random number and concatenate it to the ID, leaving the letter *"R"* as provided in the template. Although it stated in the preliminary questions that it knows today's date, when asked to use it in building the IDs, ChatGPT was easily misguided by provided examples and copied their dates, instead of the current one. Unfortunately, ChatGPT cannot be trusted yet, as for the same input, we may get quite different outputs, therefore being non-deterministic.

Regarding its usage in TOD Systems, based on the above conclusions, ChatGPT cannot be directly integrated with no further fine-tuning.

Future work may consider more detailed prompts with diverse use cases, an extended ontology, and other LLMs. In this way, we can truly test LLMs capabilities on the KGC and KGR tasks, on both Static and Temporal KGs, and their pos-sible integration with Task-Oriented Dialogue Systems.

In conclusion, our work has highlighted the advantages and disadvantages of using LLMs (here, ChatGPT) for the Static and, most importantly, Temporal KGC and KGR tasks. Our results show prominent use cases, while underlying the necessity of carefully monitoring them.

References

1. Fill, H., Fettke, P., Köpke, J.: Conceptual Modeling and Large Language Models: Impressions From First Experiments With ChatGPT. Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model. **18**, 3 (2023). https://doi.org/10.18417/EMISA.18.3

2. Han, J., Collier, N., Buntine, W.L., Shareghi, E.: PiVe: Prompting with Iterative Verification Improving Graph-based Generative Capability of LLMs. CoRR **abs/2305.12392** (2023). https://doi.org/10.48550/ARXIV.2305.12392
3. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Kirrane, S., Gayo, J.E.L., Navigli, R., Neumaier, S., Ngomo, A.N., Polleres, A., Rashid, S.M., Rula, A., Schmelzeisen, L., Sequeda, J., Staab, S., Zimmermann, A.: Knowledge Graphs. Synthesis Lectures on Data, Semantics, and Knowledge, Morgan & Claypool Publishers (2021). https://doi.org/10.2200/S01125ED1V01Y202109DSK022
4. Iga, V.I., Silaghi, G.C.: Leveraging bert for natural language understanding of domain-specific knowledge. In: 25th Intl. Symp. on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2023, Nancy, France. IEEE, to appear
5. Iga, V.I., Silaghi, G.C.: Ontology-based dialogue system for domain-specific knowledge acquisition. In: A. R. da Silva et al. (ed.) ISD2023 Proceedings, Lisboa, Portugal. AIS (2023). https://doi.org/10.62036/ISD.2023.46
6. Ji, S., Pan, S., Cambria, E., Marttinen, P., Yu, P.S.: A survey on knowledge graphs: Representation, acquisition, and applications. IEEE Trans. Neural Networks Learn. Syst. **33**(2), 494–514 (2022). https://doi.org/10.1109/TNNLS.2021.3070843
7. Liang, K., Meng, L., Liu, M., Liu, Y., Tu, W., Wang, S., Zhou, S., Liu, X., Sun, F.: A survey of knowledge graph reasoning on graph types: Static, dynamic, and multimodal. CoRR **abs/2212.05767** (2022). https://doi.org/10.48550/arXiv.2212.05767
8. Nguyen, V., Bodenreider, O., Sheth, A.P.: Don't like RDF reification?: making statements about statements using singleton property. In: Chin-Wan Chung et al. (ed.) 23rd Intl. WWW Conf., Seoul, Republic of Korea, 2014. pp. 759–770. ACM (2014). https://doi.org/10.1145/2566486.2567973
9. Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., Wu, X.: Unifying large language models and knowledge graphs: A roadmap. CoRR **abs/2306.08302** (2023). https://doi.org/10.48550/ARXIV.2306.08302
10. Wang, J., Wang, B., Qiu, M., Pan, S., Xiong, B., Liu, H., Luo, L., Liu, T., Hu, Y., Yin, B., Gao, W.: A survey on temporal knowledge graph completion: Taxonomy, progress, and prospects. CoRR **abs/2308.02457** (2023). https://doi.org/10.48550/ARXIV.2308.02457
11. Wei, X., Cui, X., Cheng, N., Wang, X., Zhang, X., Huang, S., Xie, P., Xu, J., Chen, Y., Zhang, M., Jiang, Y., Han, W.: Zero-shot information extraction via chatting with chatgpt. CoRR **abs/2302.10205** (2023). https://doi.org/10.48550/ARXIV.2302.10205
12. Zhang, J., Chen, B., Zhang, L., Ke, X., Ding, H.: Neural, symbolic and neural-symbolic reasoning on knowledge graphs. AI Open **2**, 14–35 (2021). https://doi.org/10.1016/J.AIOPEN.2021.03.001
13. Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J., Wen, J.: A survey of large language models. CoRR **abs/2303.18223** (2023). https://doi.org/10.48550/ARXIV.2303.18223
14. 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. CoRR **abs/2305.13168** (2023). https://doi.org/10.48550/ARXIV.2305.13168