

Enhancing LLMs with Knowledge Graphs for Academic Literature Retrieval

Catarina Pires¹^a, Pedro Gonçalo Correia¹^b, Pedro Silva¹^c and Liliana Ferreira^{1,2}^d

¹Faculty of Engineering, University of Porto, R. Dr. Roberto Frias, Porto, Portugal

²Fraunhofer Portugal AICOS, Rua Alfredo Allen 455/461, Porto, 4200-135, Portugal
{up201907925, up201905348, up201907523, lsferreira}@edu.fe.up.pt

Keywords: Knowledge Graphs, Large Language Models, Knowledge Graph Augmented LLMs, Academic Literature Retrieval, Natural Language Generation, Hallucination, Prompt Engineering.

Abstract: While Large Language Models have demonstrated significant advancements in Natural Language Generation, they frequently produce erroneous or nonsensical texts. This phenomenon, known as hallucination, raises concerns about the reliability of Large Language Models, particularly when users seek accurate information, such as in academic literature retrieval. This paper addresses the challenge of hallucination in Large Language Models by integrating them with Knowledge Graphs using prompt engineering. We introduce *GPTscholar*, an initial study designed to enhance Large Language Models responses in the field of computer science academic literature retrieval. The authors manually evaluated the quality of responses and frequency of hallucinations on 40 prompts across 4 different use cases. We conclude that the approach is promising, as the system outperforms the results we obtained with gpt-3.5-turbo without Knowledge Graphs.

1 INTRODUCTION

Despite the remarkable success of Large Language Models (LLMs) for Natural Language Generation in recent years, it has been shown that these models will frequently generate nonsensical or inaccurate texts, a phenomenon known as hallucination (Ji et al., 2023). These models become unreliable, as they are prone to answer a user prompt in a confident tone with incorrect information. Particularly on prompts related to academic literature, LLMs commonly refer to non-existing titles, digital object identifiers (DOI), and authors or confuse information from different publications. Given the importance of accurate information in this domain, such a response from the LLM may bring no value to the user, or, worst case, be even misleading (Emsley, 2023; Goddard, 2023).


An approach to mitigate hallucinations in LLMs is to combine them with Knowledge Graphs (KG) (Pan et al., 2023). By interconnecting typed entities and their attributes in a structured way (Pan et al., 2017), KGs may be used to inform the model or restrict its


output, preventing it from generating inaccurate information. In this paper, we explore the potential to improve LLMs' responses in the domain of computer science academic literature retrieval by making it query a KG to retrieve the relevant facts about publications. We introduce *GPTscholar*, a proof-of-concept system for natural language queries and answers in the same domain.


Section 2, presents and discusses similar solutions to reduce LLM hallucinations by leveraging KGs, which were applied to different domains. In Section 3, we explain the architecture and implementation details of the solution. In Section 4, we describe the experiment we conducted in order to evaluate the solution. In Section 5, the obtained results are presented. These results are discussed in Section 6. Finally, in Section 7, the main conclusions of the study are presented.


2 RELATED WORK

Despite the capabilities of LLMs, significant concerns have emerged regarding their propensity to generate non-factual or misleading content. This issue, known as the factuality problem, can lead to misunderstand-

^a <https://orcid.org/0009-0005-5000-6333>

^b <https://orcid.org/0009-0002-1728-0670>

^c <https://orcid.org/0009-0005-2465-2788>

^d <https://orcid.org/0000-0002-2050-6178>

ings and potentially harmful consequences, especially in domains that demand high levels of accuracy such as health, law, and finance (Wang et al., 2023).

Researchers have explored strategies to mitigate LLM hallucinations across various domains by incorporating knowledge graphs (KGs). Leveraging KGs as a source of external knowledge holds promise for enhancing LLM performance. KGs offer structured information about entities and their relationships, which aids LLMs in reasoning more effectively and significantly reduces hallucinations.

According to Agrawal, G., Kumarage, T., Alghami, Z., and Liu, H. (Agrawal et al., 2023), there are three primary approaches for leveraging Knowledge Graphs (KGs) to enhance Large Language Models (LLMs): Knowledge-Aware Inference, Knowledge-Aware Learning, and Knowledge-Aware Validation. These methodologies are distinguished by their respective stages within the retrieval pipeline architecture, particularly concerning the point at which the KG is integrated. Knowledge-Aware Inference involves incorporating KGs at the input stage to enrich the context provided to the LLM. By supplying additional relevant information, it aids the model in better comprehending the prompt, thereby reducing the likelihood of irrelevant or nonsensical outputs. Knowledge-Aware Learning focuses on embedding KGs into the training process of LLMs. Introducing factual knowledge during training enhances the model’s capacity to learn and generate accurate responses, leading to more reliable outputs. Knowledge-Aware Validation establishes mechanisms to verify the LLM’s outputs using KGs. By cross-referencing the generated content with factual information contained within the knowledge graph, this approach significantly improves the model’s reasoning and accuracy.

This study follows the first approach, Knowledge-Aware Inference, leveraging KGs at the input stage. Martino, A., Iannelli, M. and Truong, C. (Martino et al., 2023) proposed a similar approach that aimed at reducing hallucinations and improving responses to online customer reviews. To do so, information from a KG is mapped to a templated prompt that serves as input to the LLM, providing context about the place that is being reviewed. A manual evaluation process was conducted by domain experts, who rated each response and tallied the number of correct and incorrect assertions. They concluded that the KG-enhanced LLM responses had more correct assertions, fewer incorrect assertions, and better response ratings.

Brate, R. et al. (Brate et al., 2022) leverage Wikidata to improve language models on the task of movie genre classification. SPARQL queries extract movie

information from Wikidata, and the results are fed into a templated prompt to the language model. The evaluation was done on a subset of the ML25M dataset (Harper and Konstan, 2016). They concluded that the context from the KG improved the results unless too much information was given relative to the size of the language model.

The study presented is distinct from the aforementioned works in that it is the LLM that produces both the SPARQL query to the KG and the final response to the user, as well as the fact that it is applied to the domain of academic literature retrieval.

3 GPTscholar

The main idea behind the system presented here is that when a user writes a prompt, two prompts are sent to an LLM under the hood before giving the answer to the user. The first prompt queries a KG to obtain accurate information from a reliable source, while the second prompt produces an answer to the user using this information to avoid hallucinations. For the LLM, we used OpenAI’s *gpt-3.5-turbo*¹. For the KG, we used the DBLP Computer Science Bibliography (Ley, 2002), which is accessible through a SPARQL endpoint² and contains bibliographic information on major computer science publications, counting with over seven million publications and over three million authors. We devised a flow with five steps based on this idea, as illustrated in Figure 1.

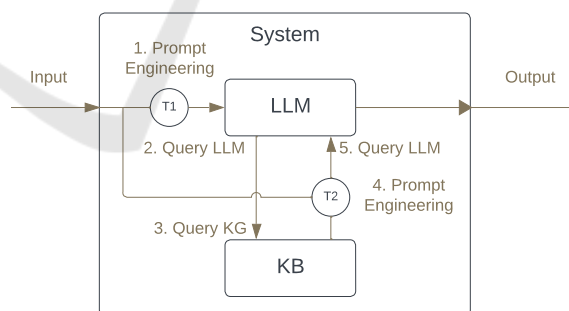


Figure 1: Flow from the initial user input to the final output, with the five steps described in Section 3. T1 and T2 represent the templated prompts to the LLM.

In the first step, the user prompt is converted into a prompt for the LLM to generate SPARQL code to get relevant information from the KG, using the template shown in Figure 2. Some instructions are given so that the LLM does not include natural language in the response, the duplicated DOIs are eliminated,

¹<https://platform.openai.com/docs/models/gpt-3-5>

²<https://sparql.dblp.org>

year operations are done in a way supported by DBLP, and some constructs that aren't supported are explicitly avoided. An example of a generic SPARQL query is also given, as well as the list of properties and publication types supported.

Create a SPARQL query to access the DBLP database and answer the given prompt. Your answer will be automatically fed to a SPARQL endpoint, so do not include any natural language text in your response. Note that there are multiple possible entries for ?doi. Only present the minimum ?doi per publication, so do not forget the GROUP BY. Note that the ?author may not have ?orcid. Note that converting ?year to an integer is not supported, so when comparing it, always compare with another string. Never include language tags such as @en or @fr in your answers, as it is not present in the database. IN and NOT IN operations are not supported. Do not substitute variables for strings directly, always use FILTER instead.

Here is an example of accessing the SPARQL endpoint:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dblp: <https://dblp.org/rdf/schema#>
```

```
SELECT ?title (min(?dois) as ?doi) ?year ?publishedIn ?
  authorName ?orcid WHERE {
  ?publication
    dblp:doi ?dois ;
    dblp:title ?title ;
    dblp:authoredBy ?author ;
    dblp:yearOfPublication ?year ;
    dblp:publishedIn ?publishedIn .
  ?author dblp:primaryCreatorName ?authorName .
  OPTIONAL { ?author dblp:orcid ?orcid }
} GROUP BY ?title ?type ?year ?publishedIn ?authorName ?
  orcid
ORDER BY DESC(?year) ?title
```

The publications only contain the following properties: dblp:title dblp:doi dblp:authoredBy dblp:publishedIn dblp:yearOfPublication rdf:type rdfs:label dblp:bibtexType dblp:numberOfCreators dblp:primaryDocumentPage dblp:pagination
The authors only contains the following properties: dblp:primaryCreatorName dblp:orcid
?type can only be one of the following: dblp:Article dblp:Inproceedings dblp:Incollection dblp:Book dblp>Data dblp:Editorship dblp:Informal dblp:Publication dblp:Reference dblp:Withdrawn
The prompt to answer is as follows:
<prompt>{user_prompt}</prompt>

Figure 2: Template for the first prompt to the LLM, which tells it to generate a SPARQL query based on the user prompt. *user_prompt* is replaced with the original user prompt.

In the second step, this prompt is fed to the LLM, and some processing is done to the response, such as stripping the markdown marks for code blocks, fixing the SPARQL prefixes if needed, and changing the query results limit to 100 to take into account that each publication has one result per author.

In the third step, the KG is queried, and its results are condensed in JSON format, where the authors of the same paper are merged into the same entry. The results are then truncated to the limit originally imposed by the LLM if any, or 10 entries to avoid exceeding the maximum tokens supported in a prompt to the LLM.

In the fourth step, these results are combined with the original user prompt using the template shown in Figure 3. The answer is asked to be in natural language, avoiding technical details such as code or references to the DBLP database. Instructions for the

answer to give when an error occurs and to include ORCID as a link on the author's name whenever possible are also given.

```
Answer the given prompt based on the results retrieved from the DBLP
database using the query you previously gave. Your answer will be sent to the
end user, so write it in natural language. Avoid writing code. Never mention
that results or errors come from DBLP database, as it is just an implementation
detail. If there is an error, apologize to the user without mentioning DBLP
database. For each author with an ORCID, make it an hyperlink on the name.
These were the results from the database:
{kb_bindings}
The prompt to answer is as follows:
<prompt>{user_prompt}</prompt>
```

Figure 3: Template for the second prompt to the LLM, which tells it to generate a response based on the KG results and the user prompt. *user_prompt* is replaced with the original user prompt, while *kb_bindings* is replaced with the results of the KG after processing.

In the final step, the resulting prompt is fed into the LLM, and its result is output to the user.

The system, named *GPTscholar*³, includes a simple web server to showcase how the solution would be used and viewed by an end user, with the frontend implemented in React⁴ and the backend in Flask⁵. This backend interfaces with the LLM and KG to reproduce the flow described in the previous paragraph.

4 EXPERIMENT

We prepared 40 prompts across 4 different use cases (10 prompts each) for academic literature retrieval to evaluate the system, using OpenAI's *gpt-3.5-turbo* without prompt engineering as a baseline. The use cases comprise the following:

1. Get publications based on the **author** (e.g. "Give me 3 papers authored by Wayne Xin Zhao.");
2. Get publications based on their **domain** (e.g. "Give me articles about generative music.");
3. Get publications based on their **attributes** (e.g. "Retrieve articles from ROBIO published before the year 2018");
4. Get publications based on **information from another publication** (e.g. "Enumerate papers written by the same authors of 'From the Semantic Web to social machines: A research challenge for AI on the World Wide Web.'").

In the experiment, our system uses OpenAI's *gpt-3.5-turbo* as the LLM and DBLP's SPARQL endpoint as the KG. We employed the schema released on October 17, 2023, which includes 61 classes, 45 object

³<https://github.com/Goncalerta/GPTScholar>

⁴<https://react.dev>

⁵<https://flask.palletsprojects.com/en/3.0.x>

Table 1: Results of the experiment for each use case. *B* denotes the baseline (gpt-3.5-turbo), while *S* denotes the GPTscholar system.

		Use Case							
		1		2		3		4	
		B	S	B	S	B	S	B	S
Prompts	Correct	2	6	1	8	2	4	0	4
	Incorrect	4	0	9	0	2	0	4	4
	No Results	4	4	0	2	6	7	6	2
	Total	10	10	10	10	10	10	10	10
Mentioned Publications	Correct	6	46	40	47	14	16	4	35
	Hallucinated Title	12	0	20	0	12	0	7	6
	Partially Incorrect	2	0	24	0	0	0	2	0
	Total	20	46	84	47	26	16	13	41

properties, and 28 datatype properties. Additionally, we used DBLP’s RDF dump from December 1, 2023, which contains a total of 378,406,765 triples, including 3,384,740 person entities, 6,972,941 publication entities, and 9,355,764 external URIs.

After running the prompts through the system and the baseline, we evaluate the results manually by analyzing each response to assess its quality and the frequency of hallucinations. For each use case, we count the number of responses that:

- correctly answered the respective prompt;
- answered the prompt with incorrect information;
- found no results.

To assess the frequency of hallucinations, the total number of publications mentioned and the number of publication titles mentioned that do not exist were evaluated, as well as the number of publications whose titles exist but have incorrect information.

5 RESULTS

The results can be found in Table 1, comparing our system, *S*, to the baseline *B*. For each use case, the table shows the number of prompts where each system gave a correct result, an incorrect result, and where it did not give any result. The table also shows the number of publications mentioned by each system in the given use case. These mentions are categorized into correct publications, publications where the title has been hallucinated and does not exist, and publications that exist, but some of the details provided by the system were incorrect.

GPTscholar significantly outperformed the baseline in every use case. In total, our system got 55% of the prompts correct, while the baseline only got 12.5%. In the first three use cases (retrieving publications based on the author, their domain, and

their attributes), the system did not hallucinate publications nor mention incorrect information in any prompt, while in the last use case (retrieving publications based on information from another publication), which requires more complex reasoning with indirect steps, it produced fewer incorrect results than the baseline. Even when GPTScholar couldn’t answer correctly, it would more likely present no results to the user than present wrong results.

6 DISCUSSION

Given that the system queries an LLM twice and a KG once, it has more points of failure than the baseline. However, our system achieved significantly better results than the baseline since it is augmented with knowledge about publications. This suggests that the task of retrieving publications without hallucinating and without consulting a KG is more difficult for state-of-the-art LLMs than the tasks of generating a SPARQL query and generating an answer based on information present in the prompt.

We manually analyzed the intermediate step from our system in the prompts with incorrect final results. All incorrect results from the system generated incorrect SPARQL code or used SPARQL operations not supported by the SPARQL endpoint, which suggests this is the intermediate step with the biggest potential for improvement.

While GPTscholar outperformed the baseline, it had the downside of being significantly slower due to querying the LLM twice and the KG once. This can lead to a worse user experience, especially since the final output can only start being written in the last query to the LLM.

7 CONCLUSION

Leveraging KGs to enhance LLMs is a promising approach to increasing the accuracy of responses and reducing hallucinations and incorrect facts. In this document, a system is introduced to retrieve academic literature information through natural language queries and responses. After the evaluation of the solution, it can be concluded that the proposed approach hallucinates less frequently than an LLM without KGs.

For future work, different prompt templates could be tried and compared, namely to improve the generation of SPARQL code. We also envision the expansion of the DBLP knowledge base to include the abstract or even the body of the publication, which would allow the LLM to answer queries that require reasoning about the content of publications. Another possibility would be the integration of knowledge bases of academic publications in fields other than computer science. Additionally, we could analyze and assess the limitations associated with having intermediate steps, as these introduce multiple points of failure. By identifying and evaluating the error potential at each stage, we can pinpoint the most critical step and focus our efforts on improving the overall system.

REFERENCES

- Agrawal, G., Kumarage, T., Alghami, Z., and Liu, H. (2023). Can knowledge graphs reduce hallucinations in llms? : A survey. *CoRR*, abs/2311.07914.
- Brate, R., Dang, M.-H., Hoppe, F., He, Y., Meroño-Peñuela, A., and Sadashivaiah, V. (2022). Improving Language Model Predictions via Prompts Enriched with Knowledge Graphs *. In *DL4KG@ ISWC2022*, Hangzhou, China.
- Emsley, R. (2023). Chatgpt: these are not hallucinations – they’re fabrications and falsifications. *Schizophrenia*, 9(1):52.
- Goddard, J. (2023). Hallucinations in chatgpt: A cautionary tale for biomedical researchers. *The American Journal of Medicine*, 136(11):1059–1060.
- Harper, F. M. and Konstan, J. A. (2016). The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19.
- Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).
- Ley, M. (2002). The dblp computer science bibliography: Evolution, research issues, perspectives. In Laender, A. H. F. and Oliveira, A. L., editors, *String Processing and Information Retrieval*, pages 1–10, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Martino, A., Iannelli, M., and Truong, C. (2023). Knowledge injection to counter large language model (llm) hallucination. In Pesquita, C., Skaf-Molli, H., Efthymiou, V., Kirrane, S., Ngonga, A., Collarana, D., Cerqueira, R., Alam, M., Trojahn, C., and Hertling, S., editors, *The Semantic Web: ESWC 2023 Satellite Events*, pages 182–185, Cham. Springer Nature Switzerland.
- Pan, J. Z., Razniewski, S., Kalo, J.-C., Singhanian, S., Chen, J., Dietze, S., Jabeen, H., Omeliyanenko, J., Zhang, W., Lissandrini, M., Biswas, R., de Melo, G., Bonifati, A., Vakaj, E., Dragoni, M., and Graux, D. (2023). Large language models and knowledge graphs: Opportunities and challenges.
- Pan, J. Z., Vetere, G., Gomez-Perez, J. M., and Wu, H. (2017). *Exploiting Linked Data and Knowledge Graphs in Large Organisations*. Springer Publishing Company, Incorporated, 1st edition.
- Wang, C., Liu, X., Yue, Y., Tang, X., Zhang, T., Jiayang, C., Yao, Y., Gao, W., Hu, X., Qi, Z., Wang, Y., Yang, L., Wang, J., Xie, X., Zhang, Z., and Zhang, Y. (2023). Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *CoRR*, abs/2310.07521.