AMPhionQA: Adaptable Modular Prompting for Translating Natural Language to SPARQL Queries

Seth J. Rosen
Wisecube AI
Bothell, WA, USA
seth.rosen@wisecube.ai

Alexander N. Thomas
Wisecube AI
Bothell, WA, USA
alex@wisecube.ai

Vishnu Vettrivel
Wisecube AI
Bothell, WA, USA
vishnu@wisecube.ai

ABSTRACT

Knowledge graphs (KGs) are useful in organizing large biomedical data sets which can be navigated efficiently via a structured query language such as SPARQL. However, requiring direct interaction with KGs restricts access to those who are familiar with graph query methods. Many available tools rely on heuristics which decrease in performance as the KG increases in size or complexity and are not designed to handle the multi-hop biomedical questions posed by end users. We present AMPhionQA, a novel solution for natural language (NL) querying of KGs using adaptable modular prompting to improve SPARQL construction for information retrieval. AMPhionQA incorporates indexed question-response pairs to create large language model (LLM) prompts capable of processing a broader range of user inputs to generate more accurate intermediate representations for translation into SPARQL queries. We mitigate training time and costs without compromising performance by dynamically generating LLM prompts at runtime, incorporating examples that closely resemble the input and thereby surpassing the capabilities of static prompts. Our approach caters to users more inclined towards contextual applications of information retrieval, prioritizing the broader picture over specialized knowledge outside of traditional graph querying. By utilizing pre-trained LLMs to facilitate the translation of NL to SPARQL, AMPhionQA enhances accessibility to KGs and structured data for users with biomedical expertise. We provide a demonstration found https://github.com/wisecubeai/amphionqa_demo.

CCS Concepts

• **Information Systems → Data management systems → Database management system engines → Database query processing**

Keywords

Natural language processing; knowledge graphs; large language models; prompt tuning; prompt engineering

1. INTRODUCTION

Knowledge graphs (KGs) are useful for organizing large biomedical data sets in a structured manner by using predicates to represent common relationships between entities [1-3]. KGs have been utilized for a wide range of biomedical applications including ontologies [4-6] and drug discovery [7-9]. However, direct interaction with KGs traditionally requires an understanding of programming and graph query methods beyond that of most clinicians and biomedical researchers [10]. Additionally, users must understand the schema of a particular KG to be able to effectively retrieve information. These barriers to entry pose major limitations for implementing more widespread use of KGs for biomedical applications. Ideally, researchers should be able to pose natural language (NL) questions and retrieve info/answers from KGs without using tedious graph query methods such as SPARQL.

There have been various attempts to facilitate KG interaction through traditional machine learning (ML) approaches and visual query builders [11, 12]. Traditional ML approaches tend to use a combination of transformers and heuristics to generate SPARQL; these systems typically perform well on simple queries and/or less complex domains but struggle with harder questions [13]. By contrast, visual/heuristic approaches permit the user to select entities and predicates for the SPARQL query, allowing for the construction of more complex queries but requiring a solid understanding of the KG domain and layout [14]. Due to the complex relationships between these types of entities, biomedical users are more likely to submit multi-hop queries to gain biologically relevant insights [13, 15]. Thus, existing methods are not sufficient to cover the breadth of questions that researchers will likely need to address.

Recently, the integration of large language models (LLMs) with KGs has been presented as a solution to improve question-answering (QA) systems [16-18]. These systems typically use subgraphs as an additional input to an LLM call to provide the necessary information to correctly answer a question [19]. These successes often rely on some form of prompt tuning such as chain-of-thought prompting, which helps avoid hallucinations by forcing an LLM to make logical progressions in its answering [20]. However, chain-of-thought prompting is not effective for entity extraction from graphs because it lacks knowledge of the schema and cannot avoid hallucinations that stem from incomplete training data. Thus, LLMs have difficulty directly translating from NL to SPARQL due to the need for domain-specific schema and uniform resource identifiers (URIs) [21]. Furthermore, the cost of training an LLM to interact with a specific KG quickly becomes expensive to build and maintain [22].

We designed AMPhionQA to facilitate easy KG interaction via NL queries so end users can benefit without knowledge of relevant programming languages like SPARQL. AMPhionQA avoids common LLM problems by converting from NL questions into an intermediate representation that can more easily be converted to SPARQL. Our novel method of adaptable modular prompting (AMP) of LLMs to produce custom intermediate representations allows the system to seamlessly integrate new examples to scale and adapt to new user requests without needing to retrain a model or pause the service. By using AMP to provide relevant examples for prompt tuning, the LLM does not require

Conference'10, Month 1–2, 2010, City, State, Country.

DOI: http://dx.doi.org/10.1145/12345.67890

knowledge of the graph schema for conversion of NL to intermediate representations. AMPhionQA is then able to conduct entity linking and construct SPARQL queries to retrieve the relevant information from the graph. Our process allows any individual to input and retrieve information in NL without ever thinking about SPARQL or KG schema and without sacrificing accuracy. AMPhionQA makes KGs more accessible and valuable to clinicians and biomedical researchers.

2. AMPHIONQA

2.1. Dataset construction

Nearly all existing biomedical datasets for KG QA either expect a generated text answer or address a specialized area of study. Our goal for AMPhionQA was to build a system that was able to handle a wide range of biomedical topics that may be presented in various formats. To test the capabilities of AMPhionQA, we constructed our own preliminary dataset consisting of 4085 questions that encompassed approximately 1800 unique entities, 13 relationships, and several hundred different ways of phrasing questions ranging in formality. To rapidly generate examples, we utilized templates of question phrasings into which different entity labels could be inserted. Our current dataset serves as a foundation composed of relatively simple questions on topics we anticipate will be more frequently queried by biomedical users or that are common components of more complex queries. We plan to add more diverse and complex examples as we continue to develop the system and gain additional feedback from users.

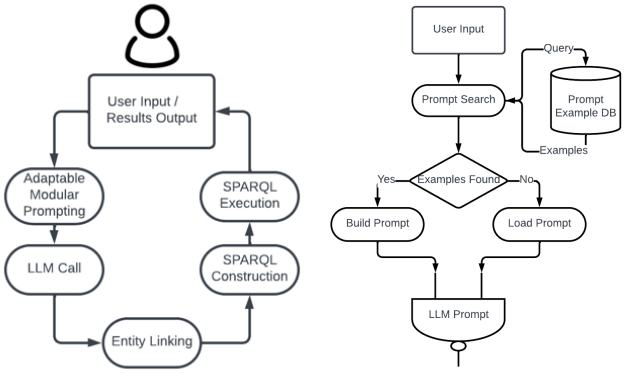

Figure 1. Overview of AMPhionQA system (left) and adaptable modular prompting (right).

2.2. Adaptable modular prompting of large language models

Prior attempts to integrate LLMs and KGs for biomedical applications using generalized static prompts have proven inadequate in providing a comprehensive QA solution, primarily due to their tailored design for specific use cases. This limitation arises from the inherent focus of static prompts on users with inquiries confined to particular contexts or subjects, rendering them ineffective in accommodating broader information retrieval needs. Employing multiple static prompts can broaden the scope of topics addressed by a system; however, this approach requires extra logic and/or LLM calls to determine the optimal prompt for the input. Instead, AMPhionQA uses adaptable modular prompting (AMP) to increase the capabilities of pre-trained LLMs to complete NL-KG interactions.

AMPhionQA begins by taking the user's input question and cross-referencing it with stored examples in the example prompt database to identify examples with similar components, which are then used to prime the LLM prompt (Fig. 1). If no similar examples are available, a fallback general prompt is utilized. Once the prompt is generated via AMP, an LLM call is made to convert the user's input into a customized intermediate representation (IR) format required for entity linking and SPARQL generation. This IR schema is intentionally designed to be domain-agnostic, facilitating easy adaptation to diverse domains and reducing the risk of errors attributable to insufficient domain-specific training data in the LLM training corpus. Following the return of the intermediate representation by the LLM, entity linking is performed to bind the IR to our KG schema by accurately determining the URIs for the provided entity and predicate labels. The intermediate representation with URIs is then used to generate the corresponding SPARQL query for the NL question to retrieve information from the KG. Currently, this information is output as natural language in a table format.

Q: What are some common symptoms of influenza?
AMP examples
What are the known causes of influenza?
What are the major symptoms of cat-scratch disease?
List causes of influenza.
What are the characteristic symptoms of lupus nephritis?
What are the main clinical symptoms of toxoplasmosis?

Figure 3. Examples for priming LLM call using adaptive modular prompting (AMP). AMPhionQA takes natural language queries by the user and identifies examples with similar components in our prompt dataset. 5 to 20 examples are then provided in the LLM prompt followed by the user's original question.

3. RESULTS

3.1. Dataset splitting

We split 4085 questions into a set of 3050 for experimentation and used the remaining 1035 as AMP examples to yield a strong mix of phrasings and entities for each of the relationship groups. For each of the 3050 experimental questions, we generated corresponding SPARQL queries and expected results. For the 1035 AMP examples, we generated the corresponding intermediate representations. For our evaluations, we only returned results that corresponded to the variable of interest and excluded all other variables in cases where more than one variable was required for the SPARQL query.

Q: What are some common symptoms of influenza?		
Expected answer	**Returned answers**	
	#1	**#2**
Headache	Headache	Headache
Fatigue	Fatigue	Fatigue
Cough	Cough	Cough
Fever	Fever	Fever
Rhinitis	Chills	Rhinitis
Myalgia		Myalgia
Chest pains		Chest pains
Chills		Chills
Nasal congestion		Nasal congestion
		Nausea
		Vomiting

Figure 4. Example AMPhionQA outputs compared to expected results. Returned answers #1 and #2 highlight that different results are not necessarily incorrect. We assessed performance based on similarity scores to avoid overly penalizing partially correct results.

3.2. Metrics

To evaluate the performance of AMPhionQA, we measured the Jaccard similarity score between the expected and observed results for the final NL output. This metric considers missing or potentially extraneous entities without marking results completely incorrect. Figure 4 shows examples of potential results compared to an exact answer. Answer #1 only returned a subset of the common symptoms, while answer #2 returned all symptoms plus an additional two symptoms that are only common in children. Neither returned answer is technically incorrect, so it would be inappropriate to discard either completely, but both answers differ from the expected answer and should be scored accordingly. Hence, this approach provides a more accurate representation of whether AMPhionQA outputs a reasonable answer for a given question than simply assessing whether it returns a particular set of results, since questions can potentially be vague and/or addressed by different SPARQL queries depending on user preference. Additionally, this metric is useful in determining weaker areas for our system as well as potential problems in how we have defined what the correct answer should be.

3.3. Results

We conducted five rounds of testing using OpenAI's GPT-3.5 and an experimentation set comprising 3050 questions to evaluate the impact of varying the number of relevant examples included in the LLM prompt on performance in comparison to our previously developed static prompt. Prompts containing 5, 10, 15, and 20 examples were utilized for a given user-provided question to ascertain the optimal number of examples. The static prompt included at least one example for all groups tested.

Figure 5 illustrates that overall performance using AMP surpassed that of the all-encompassing static prompt, as well as highlighting the impact of varying the number of examples included in the prompt. AMPhionQA achieved an average similarity score of approximately 83% across all groups with anywhere from 5-20 examples provided, whereas the static prompt only attained a score of approximately 66%. Notably, the disparities become more pronounced when examining performance across different relationship groups. Certain groups (e.g., Pred9, Pred13), exhibited poor performance with the static prompt due to the limited number of examples available. AMPhionQA demonstrated dramatically superior performance in these instances because it incorporated relevant questions composed of similar components regardless of their predicate groups. While not explicitly documented, we observed a significant slowdown in runtime between the runs utilizing 15 and 20 examples in the prompt, indicating potential scalability concerns with larger prompt sizes. Thus, we propose that 10 examples are sufficient for AMPhionQA to accurately return relevant results.

4. CONCLUSIONS

Here, we introduce AMPhionQA, a new NL knowledge graph query tool tailored for biomedical sciences. Powered by our novel adaptive modular prompting (AMP) technique, AMPhionQA surpasses the limitations of conventional static prompts, exhibiting superior performance while enabling rapid scalability without the need for extensive retraining or redeployment. Our preliminary evaluations have demonstrated AMPhionQA's proficiency in enhancing LLM prompt generation, thereby refining the accuracy of transforming natural language to SPARQL queries. Moreover, our findings underscore its potential applicability beyond biomedical tasks, particularly in addressing diverse NL challenges characterized by high variability and associated training costs.

By leveraging the capabilities of LLMs while mitigating common challenges, AMPhionQA represents a significant advancement in natural language interaction with knowledge graphs. By employing adaptable modular prompting, we not only enhance LLM performance but also mitigate the risk of hallucinations, thus bolstering the tool's reliability and usability.

Our commitment to democratizing data access is evident in our pursuit of techniques that streamline the training process, ensuring cost-effectiveness without sacrificing performance. Furthermore, our emphasis on refining evaluation metrics reinforces our dedication to continuous improvement and transparency.

Looking ahead, we aim to further augment AMPhionQA's functionality to offer users a more personalized experience, allowing them to tailor their query results to meet specific requirements. This entails expanding its repertoire to handle a broader range of questions, including more complex inquiries, and offering alternative output formats such as text-based answers.

In conclusion, AMPhionQA showcases how advanced language models and innovative prompting techniques can work together, providing a preview of future information retrieval across various fields that is both accessible and efficient.

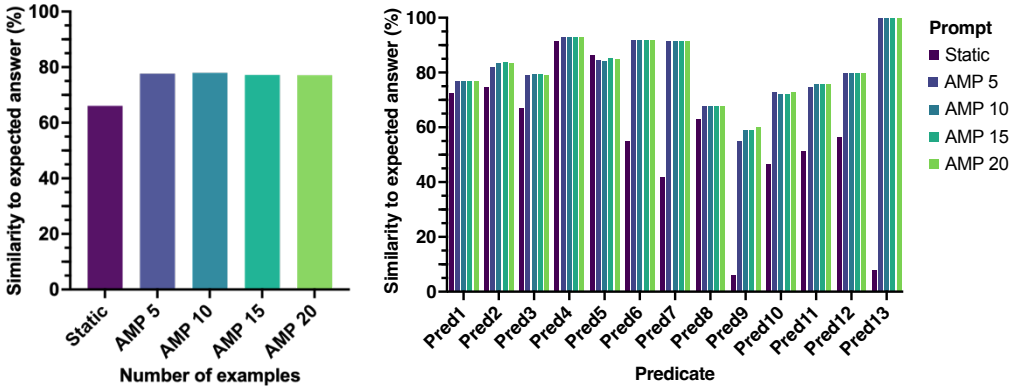

Figure 5. Accuracy of AMPhionQA vs. static prompts for generating answers to NL questions from KG data. Average similarity scores are shown overall for the whole run regardless of predicate group (top) and by predicate group bottom). Some predicate group exhibited poor performance with the static prompt due to a limited number of examples but still performed well using AMP. Similarity scores were consistent across AMP groups regardless of number of prompts. AMP 5-20 labels refer to the number of examples provided from the prompt database.

5. ACKNOWLEDGMENTS

6. REFERENCES

[1] Nicholson, D. N. and Greene, C. S. Constructing knowledge graphs and their biomedical applications. 2020. *Computational and Structural Biotechnology Journal*, 18 (2020/01/01/ 2020), 1414-1428.

[2] Chang, D., Balazevic, I., Allen, C., Chawla, D., Brandt, C. and Taylor, R. A. Benchmark and Best Practices for Biomedical Knowledge Graph Embeddings. 2020. *Proc Conf Assoc Comput Linguist Meet*, 2020 (Jul 2020), 167-176.

[3] Yuan, J., Jin, Z., Guo, H., Jin, H., Zhang, X., Smith, T. and Luo, J. Constructing biomedical domain-specific knowledge graph with minimum supervision. 2020. *Knowledge and Information Systems*, 62 (2020), 317-336.

[4] Huang, Z., Hu, Q., Liao, M., Miao, C., Wang, C. and Liu, G. Knowledge Graphs of Kawasaki Disease. 2021. *Health Inf Sci Syst*, 9, 1 (Dec 2021), 11.

[5] Kurbatova, N. and Swiers, R. Disease ontologies for knowledge graphs. 2021. *BMC Bioinformatics*, 22, 1 (Jul 21 2021), 377.

[6] Silva, M. C., Eugenio, P., Faria, D. and Pesquita, C. Ontologies and Knowledge Graphs in Oncology Research. 2022. *Cancers (Basel)*, 14, 8 (Apr 10 2022).

[7] Bonner, S., Barrett, I. P., Ye, C., Swiers, R., Engkvist, O., Bender, A., Hoyt, C. T. and Hamilton, W. L. A review of biomedical datasets relating to drug discovery: a knowledge graph perspective. 2022. *Briefings in Bioinformatics*, 23, 6 (2022).

[8] MacLean, F. Knowledge graphs and their applications in drug discovery. 2021. *Expert Opin Drug Discov*, 16, 9 (Sep 2021), 1057-1069.

[9] James, T. and Hennig, H. Knowledge Graphs and Their Applications in Drug Discovery. 2024. *Methods Mol Biol*, 2716 (2024), 203-221.

[10] Callahan, T. J., Tripodi, I. J., Pielke-Lombardo, H. and Hunter, L. E. Knowledge-Based Biomedical Data Science. 2020. *Annu Rev Biomed Data Sci*, 3 (Jul 2020), 23-41.

[11] Yu, D., Zhang, S., Ng, P., Zhu, H., Hanbo Li, A., Wang, J., Hu, Y., Wang, W., Wang, Z. and Xiang, B. 2022. *DecAF: Joint Decoding of Answers and Logical Forms for Question Answering over Knowledge Bases*. City, 2022.

[12] Pradel, C., Haemmerlé, O. and Hernandez, N., Jane. 2013. *Natural language query interpretation into SPARQL using patterns*. City, 2013.

[13] Yin, X., Gromann, D. and Rudolph, S. 2019. *Neural Machine Translating from Natural Language to SPARQL*. City, 2019.

[14] Francart, T. 2023. *Sparnatural: a visual knowledge graph exploration tool*. Springer, City, 2023.

[15] Peng, C., Xia, F., Naseriparsa, M. and Osborne, F. Knowledge Graphs: Opportunities and Challenges. 2023. *Artif Intell Rev* (Apr 3 2023), 1-32.

[16] Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J. and Wu, X. Unifying Large Language Models and Knowledge Graphs: A Roadmap. 2024. *IEEE Transactions on Knowledge and Data Engineering* (2024), 1-20.

[17] Song, Y., Li, W., Dai, G. and Shang, X. Advancements in Complex Knowledge Graph Question Answering: A Survey. 2023. *Electronics*, 12, 21 (2023), 4395.

[18] Yang, L., Chen, H., Li, Z., Ding, X. and Wu, X. Give Us the Facts: Enhancing Large Language Models with Knowledge Graphs for Fact-aware Language Modeling. 2024. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[19] Martino, A., Iannelli, M. and Truong, C. 2023. *Knowledge Injection to Counter Large Language Model (LLM) Hallucination*. Springer, City, 2023.

[20] Nori, H., Lee, Y. T., Zhang, S., Carignan, D., Edgar, R., Fusi, N., King, N., Larson, J., Li, Y., Liu, W., Luo, R., McKinney, S. M., Osazuwa Ness, R., Poon, H., Qin, T., Usuyama, N., White, C. and Horvitz, E. 2023. *Can Generalist Foundation Models Outcompete Special-Purpose Tuning? Case Study in Medicine*. City, 2023.

[21] Soman, K., Rose, P. W., Morris, J. H., Akbas, R. E., Smith, B., Peetoom, B., Villouta-Reyes, C., Cerono, G., Shi, Y. and Rizk-Jackson, A. Biomedical knowledge graph-enhanced prompt generation for large language models. 2023. *arXiv preprint arXiv:2311.17330* (2023).

[22] Vu, T., Iyyer, M., Wang, X., Constant, N., Wei, J., Wei, J., Tar, C., Sung, Y.-H., Zhou, D., Le, Q. and Luong, T. 2023. *FreshLLMs: Refreshing Large Language Models with Search Engine Augmentation*. City, 2023.