Drug Discovery Today

Natural Language Querying of Biological Databases with Large Language Models

--Manuscript Draft--

Manuscript Number:	DRUDIS-D-25-00245
Article Type:	Reviews 2025 - II
Keywords:	Artificial Intelligence; Machine Learning; Large Language Model; Pharmaceutical; Drug Discovery; Life Sciences
Abstract:	It is attractive to be able to query biological knowledge bases using queries written in a natural language. Most recently a number of high-profile "AI co-scientist" projects were announced that demand extremely accurate natural language querying. Natural language queries are typically translated into structured queries using Large Language Models (LLM). However, unassisted naive LLMs often fail to interpret the scientific queries correctly and also may produce linguistically correct but factually false output colloquially known as hallucinations. Here we review the current practices in natural language querying with LLMs, and report the outcomes of a systematic assessment of these practices. We find that the best balance between accuracy and flexibility in this context is achieved by multiple LLM agents that can challenge the outputs of each other and interact with a human user. This strategy is very flexible and does not require prior knowledge of the data structure, query templates, secondary databases, or adaptor language models. We highlight the need for appropriate benchmarks for the assessment of natural language data mining systems, and provide a forward-looking view on how to best structure the application of LLMs to natural language querying.

Highlights

- Accurate natural language data mining is a requirement for co-scientist AI systems
- LLM agents in combination with deterministic queries offer the best accuracy
- LLM agents must be Findable and Reusable (FAIR) and thus require API standards
- Benchmarks for testing of natural language data mining systems are needed

Natural Language Querying of Biological Databases with Large Language Models

Authors

- Vladimir A. Makarov, Pistoia Alliance, Wakefield, MA 01880 USA; vladimir.makarov@pistoiaalliance.org (corresponding author),ORCID 0000-0002-8218-1145
- Oleg Stroganov, Rancho Biosciences, Rancho Santa Fe, CA 92067, USA; oleg.stroganov@ranchobiosciences.com, ORCID 0000-0001-6190-823X
- Laura I. Furlong, MedBioInformatics Solutions; Barcelona, 08007 Spain; laura.furlong@disgenet.com, ORCID 0000-0002-9383-528X
- Brian Evarts, Crown Point Technologies, Columbia, MD 21046 USA; brian.evarts@crownpoint.tech, ORCID 0009-0005-5146-7529
- Loes van den Biggelaar, The Hyve, Utrecht, The Netherlands; loes@thehyve.nl, ORCID 0009-0005-3790-3056
- Alexandros Goulas, Abbvie Germany; alexandros.goulas@abbvie.com , ORCID 0000-0002-5381-0096
- Etzard Stolte, Roche; etzard.stolte@roche.com, ORCID 0009-0006-3166-563X
- Derek Marren, AstraZeneca; derek.marren@astrazeneca.com, ORCID 0009-0008-1891-4261
- Lars Greiffenberg, AbbVie Germany; lars.greiffenberg@abbvie.com, ORCID 0009-0008-9052-2297

Abstract

It is attractive to be able to query biological knowledge bases using queries written in a natural language. Most recently a number of high-profile "AI co-scientist" projects were announced that demand extremely accurate natural language querying. Natural language queries are typically translated into structured queries using Large Language Models (LLM). However, unassisted naive LLMs often fail to interpret the scientific queries correctly and also may produce linguistically correct but factually false output colloquially known as hallucinations. Here we review the current practices in natural language querying with LLMs, and report the outcomes of a systematic assessment of these practices. We find that the best balance between accuracy and flexibility in this context is achieved by multiple LLM agents that can challenge the outputs of each other and interact with a human user. This strategy is very flexible and does not require prior knowledge of the data structure, query templates, secondary databases, or adaptor language models. We highlight the need for appropriate benchmarks for the assessment of natural language data mining systems, and provide a forward-looking view on how to best structure the application of LLMs to natural language querying.

Keywords

Artificial Intelligence, Machine Learning, Large Language Model, Pharmaceutical, Drug Discovery, Life Sciences

Effective querying of bioinformatics data resources [1] requires specialized knowledge, and therefore is a labor-intensive process. Using natural language for it may improve research efficiency and is also a prerequisite for any large-scale automation of scientific hypothesis generation in the "AI co-scientist" systems [2,3]. Natural language data mining is achieved by Large Language Models (LLMs) that can generate complex structured queries from natural language requests, a very active research area [4].

However, LLMs are prone to producing plausibly looking but factually false output, frequently referred to as "hallucinations" [5,6], at unacceptably high rates (up to 81% [7], and 91% for some models [5]), and that some authors see as unavoidable [8]. On the flip side, "out-of-the-box" LLMs do not have access to the newest scientific publications or proprietary knowledge that was not present in their training. These are critical issues that make it necessary to use special methods to connect the output of LLMs to the underlying facts. Many such methods have been proposed, however, at this time there is no clear grasp of the tradeoffs and payoffs of each solution.

Strategies for Natural Language Querying with LLMs

For the review of early work please refer to [9]. Many of these techniques have low accuracy either in the translation from natural language to structured query language (SQL, SPARQL, or Cypher) or in generation of the final answer [10-12].

More sophisticated techniques, such as using multiple AI agents [13-15], chain-of-thought [16], or ontologies [17] result in higher, but still imperfect accuracy. The most accurate and straightforward but the least flexible technique restricts the role of LLM to completion of a template [181] and thus sacrifices flexibility for accuracy. Currently the most popular Retrieval-Augmented Generation (RAG) techniques utilize a secondary vector database, and flatten the contents of the data sources into that database, and then retrieve relevant information from it [19,20].

Knowledge Graph RAG (KG-RAG) offers advantages over conventional RAG. RAG performs very well when questions require detailed information involving direct, "one-hop" relations between entities, while KG-RAG performs better when questions require information involving indirect, "multi-hop" relations between entities or the synthesis of information dispersed across different documents [21,22]. Consequently, different methods to build a KG-RAG system have been developed, for instance, community-based graph querying [22] and tailored KG-RAG methods for the medical domain, for example, Medical Graph RAG [23]. However, these methods involve construction of a knowledge graph in a specific way, thus limiting its use for the existing knowledge graphs.

RAG and KG-RAG methods achieve an answering accuracy above 90% [19,20] in single-shot experiments. But although these performance figures look high, it is **important to realize that all natural language query methods ultimately compete with structured queries created by hand. Such manual structured queries do not introduce any natural language to structured query language (NL2QL) translation error at all. Therefore, a completely accurate NL2QL translation is an absolute requirement.**

Testing Methodology

In this study we present a systematic evaluation of different NL2QL strategies for querying a biological knowledge graph database. Target discovery and validation was picked as the initial use case because it is a common process in pharmaceutical R&D that requires mining, integration and analysis of large and diverse datasets. We used the Open Targets Platform [24] as a source of test data. Using the ideas from

the literature review plus the recommendations of our author group we collected 21 different strategies for natural language querying. These strategies are described in Table 1. We used five different LLMs (gpt-4o-2024-08-06 (abbreviated "gpt-4o" in the subsequent sections), claude-3-5-sonnet-20240620, open-mistral-7b, o1-preview-2024-09-12 (abbreviated "o1"), and gpt-4o-mini). We included both proprietary LLMs and open-source ones. Note that not all combinations of LLMs and strategies are possible; for instance, not all frameworks support LLM agents natively. Consequently, not all combinations of LLMs and strategies were tested. We used two sets of test questions. One, an internally-developed set of three complex scientific questions that require lists of items (genes and diseases) as answers; see Table 2. Another one is a modified set of 100 true/false questions derived from the BioMix question set published earlier [19], available in the Supplement.

Results

Results for the small set of complex questions are summarized in Table 3. Results for the modified BioMix set of True/False questions are in Table 4 available in the Supplement.

Uneven model performance. The accuracy in each strategy of smaller models was uniformly less than bigger models. Even the adapters and fine-tuned versions of smaller models specifically trained on Cypher structured query generation fail in our use case.

Strategies that rely on the data schema supplied in the prompt do not work well. Contrary to our expectations, and regardless to how exactly the schema is supplied, such strategies were not successful. This includes "Naïve with DB description" (natura language database description supplied in the prompt), "Naïve with automatically generated schema", and "Naïve with GraphQA library" strategies. Enhancing the schema with example values for each node and relationship property did not improve the outcome.

The specialized Text2Cypher [11] LLM failed in all test cases, regardless of the model size (4 to 16 bit variants).

BioChatter shows poor accuracy. BioChatter [25], a built-in NL2QL mechanism in the BioCypher database, uses iterative construction of Cypher queries by analyzing entities, relationships, and node properties in the knowledge graph in succession, before attempting to generate the query itself, followed by an optional reflection agent scoring the input and allowing for the self-improvement of the query. If the reflection agent is not invoked, this strategy is similar to the automated schema generation, but distinct in its iterative approach. If the reflection agent is used, then the BioChatter should behave similarly to the agentic strategies (more on these below), but this is only supported for the gpt-4o model. Both variants failed in our experiments with the typical errors.

Entity recognition is the main source of error. A mismatch between the entity names in the prompt and in the data source (e.g. 'ALS' and 'Amyotrophic Lateral Sclerosis') can happen quite easily. Three possible solutions may be used to address this issue: applying ontologies, although this may limit introduction of new entities; entity recognition models, like BELHD [26]; and conversational agents, to clarify the meaning and context of the terms supplied in the NL query.

Other usual failure modes are errors due to the misinterpretation of the data schema, query directionality, and mistakes in the Cypher code (punctuation marks, excessive decoration of the queries). Our knowledge graph schema is based on BioCypher's Open Targets adapter, which itself is still a

prototype. Using data schemas optimized for automated querying and progressively better training of LLMs on the Cypher query language may improve the outcomes. Further details on the frequencies of different error types are available in the Supplemental files.

Task avoidance and background knowledge in LLMs. Strategies with prompt optimization with specialized libraries (DSPy [27] and MIPROv2 [28]) in some cases resulted in task avoidance behavior. LLMs would by-pass writing a query and instead answer the question based on background information stored in the LLM itself. This brought up the need to separately evaluate the LLM background knowledge and their ability to produce correct Cypher queries. Consequently, we introduced additional strategies named "Background knowledge" and "Query only" that were tested in multiple variants of prompt optimization. (Details available in the Supplement). We found that despite an apparent ability to guess the correct answers from its background information stored in the model, the LLMs cannot reliably produce structured queries, even with prompt optimization. Template or agent strategies are required for this.

Background knowledge stored in LLMs contaminates the testing results. Besides the risk of hallucinations, LLMs do not have access to any data beyond the data used when training the LLM, which means no fresh data and no proprietary data will be used for the answer. The data used to train the LLMs are not documented sufficiently well, which makes designing independent test sets impossible. Further, there are legal uncertainties in relation to the intellectual property over data used to train LLMs [29].

Tradeoffs of accuracy and flexibility. Template-based strategies achieve 100% performance in current evaluation, but require hard-coded templates for information retrieval, and cannot be transferred to another data structure without an extensive human intervention, which in turn defies the purpose of natural language querying. GPT o1 with examples of queries also achieves high accuracy, but in effect belongs to the same templated strategy type. On the other hand, the multiple-agent-based strategy achieves 83-98% performance in case of the best performing LLM (o1), and does not require prior knowledge of the data structure. It is adaptable to other databases or updates of existing ones. But it is not absolutely accurate.

Non-deterministic nature of LLMs. We observed that LLMs may produce different queries starting from the same natural language question. While semantically correct, these queries may include assumptions on the limit of the number or order of items to return. These variations are not deterministic. As a result, on different execution rounds the same natural language question may result in different answers. It is necessary to explicitly formulate the limits, order restrictions, and other parameters when asking the question, or to use a conversational agent to determine the user's intentions. A related matter, is whether specifics in the implementation of vector database in RAG may introduce implicit restrictions on what data is explored by the LLM and what data is not, and thus bias the answers. This may happen without the user understanding the restrictions, and perhaps even without the system's authors knowing that they embedded such restrictions in the system architecture.

The structure of the knowledge graph database used for queries matters. The schema for our experiment is deliberately simple, but is still a source of errors, for instance, in query directionality. Additional testing should be done to track the accuracy of the NL2QL translation based on the complexity of the schema. Moreover, schemas are defined in a structured way, while LLMs work best on the unstructured text. Future research could include variation in the schema as a parameter for testing, that may eventually lead to the ability to extract knowledge graphs directly from literature in an automated manner, and with structure optimal for querying.

The form of the prompt matters. We observed that LLMs can better produce meaningful answers from prompts that resemble a story, rather than a dry question, even if the details of the story are irrelevant to the main question asked. Automated prompt optimization strategies with DSPy [27] and MIPROv2 [28] libraries exploit this property.

LLM agents. Multi-agent-based strategy returns the highest accuracy after the template-based strategy, suggesting the best alternative to avoid hard coding and maintaining flexibility. Agentic strategy, however, comes at a higher cost. It uses more tokens because of multiple interactions between agents. It uses more user time to evaluate due to the need to engage with the user-facing agent and answer its questions, and due to the need to carefully understand each answer and evaluate its logic. Errors are more subtle to classify. That said, a strategy with multiple agents that are able to question each other's outputs and thus at least partially mimic the process of development of scientific thought seems very promising [30]. Agent frameworks, such as LangGraph, LlamaIndex Workflows, or Microsoft AutoGen may be used to orchestrate multi-agent systems. Our agent strategy can be further improved by the additional specific quality control steps (e.g. asking a specialized agent why a query did not work in case of errors), or by combining with template-based strategies (to streamline simple queries), or by accumulating lessons learned from interactions with a particular data source(s). In the future one can easily envision entire ecosystems of AI agents working together. Creation of such future AI agent ecosystems is of course dependent on the standardization of protocols for discovery of AI agent-driven services and engagement with them.

Conclusions

There are three major forward-looking conclusions from our study:

First, we acknowledge that no NL2QL strategy is absolutely error-free. Multiple-agent solutions that mimic interactions between human experts are the most promising strategy for this use case. The agentic strategy does not require prior knowledge of the database schema and thus is extremely flexible, and can be further improved by extra QC steps. The winning practical strategy for natural language data mining should include a combination of LLM agents (that can challenge the outputs of each other and interact with a human user) for complex tasks like strategy planning, and deterministic API calls to the data resources for simple data retrieval tasks that do not require AI. Taken to the extreme, this thinking leads us to a requirement for an adaptable data architecture that learns from both the newly acquired data and from newly received questions, and evolves in time to better serve the human users. Prototypes like this are already being pioneered by the research community [3].

Second, we need an industry benchmark for testing data retrieval systems from complex biological data sources. The absence of such benchmarks creates a barrier to development of NL2QL LLMs, related prompt optimizations systems, and similar methods. The benchmark should cover the individual steps of understanding the natural language question, disambiguation of terms, building the structured query, and quality control of the answer. The testing process should be resilient to the background knowledge in LLMs that may potentially contaminate the results. We are happy to see that the community is already taking steps in this direction [18, 31].

Third, we establish an emergent requirement for a set of standards for discovery and engagement with service AI agents for biological sciences. Such agents must be made Findable, Accessible, Interoperable, and Reusable (FAIR) and thus require API standards, perhaps similar to those used in Service-Oriented

Architectures today. These standards will be very important when multiple commercial entities enter the field. An example of such a standard is the Model Context Protocol [32], but additional ones may be necessary.

References

1. Rigden, D.J, Fernández, X.M., The 2025 Nucleic Acids Research database issue and the online molecular biology database collection, *Nucleic Acids Research*, **53**, Issue D1, pp. D1–D9 (2025). https://doi.org/10.1093/nar/gkae1220
2. Gottweis, J., Weng, W.H., Daryin, A., Tu, T., Palepu, A. *et al.* Towards an AI co-scientist. *arXiv preprint* arXiv:2502.18864 (2025). https://doi.org/10.48550/arXiv.2502.18864
3. Huang, K. *et al.* Biomni: A General-Purpose Biomedical AI Agent. bioRxiv 2025.05.30.656746; https://doi.org/10.1101/2025.05.30.656746
4. Lavrinovics, E., Biswas, R., Bjerva, J., Hose, K. Knowledge Graphs, Large Language Models, and Hallucinations: An NLP Perspective. *Journal of Web Semantics*, **85**, 100844 (2025). https://doi.org/10.1016/j.websem.2024.100844
5. Chelli, M. *et al.* Hallucination Rates and Reference Accuracy of ChatGPT and Bard for Systematic Reviews: Comparative Analysis. *J Med Internet Res*, **26**, e53164 (2024). https://doi.org/10.2196/53164
6. Zhou, L. *et al.* Larger and more instructable language models become less reliable. *Nature* 634, 61–68 (2024). https://doi.org/10.1038/s41586-024-07930-y
7. Wei, J. *et al.* Measuring short-form factuality in large language models. *arXiv preprint* arXiv:2411.04368 (2024). https://doi.org/10.48550/arXiv.2411.04368
8. Banerjee, S., Agarwal, A., Singla, S. LLMs Will Always Hallucinate, and We Need to Live with This. In: Arai, K. (eds) Intelligent Systems and Applications. IntelliSys 2025. Lecture Notes in Networks and Systems, vol 1554. Springer, Cham. (2025). https://doi.org/10.1007/978-3-031-99965-9_39
9. Amugongo LM, Mascheroni P, Brooks S, Doering S, Seidel J (2025) Retrieval augmented generation for large language models in healthcare: A systematic review. PLOS Digital Health 4(6): e0000877. https://doi.org/10.1371/journal.pdig.0000877
10. Kosten, C., Nooralahzadeh, F., Stockinger K. Evaluating the effectiveness of prompt engineering for knowledge graph question answering. *Frontiers in Artificial Intelligence*, **7-2024** (2025). https://doi.org/10.3389/frai.2024.1454258
11. Ozsoy, M.G., Messallem, L., Besga, J. and Minneci, G. Text2Cypher: Bridging Natural Language and Graph Databases. *arXiv preprint* arXiv:2412.10064 (2024). https://doi.org/10.48550/arXiv.2412.10064
12. Amrani, M.C., Hamouda Sidhoum, A., Mataoui, M., Baghdad Bey, K. Leveraging Large Language Models and Knowledge Graphs for Advanced Biomedical Question Answering Systems. In: Djamaa, B., Boudane, A., Mazari Abdessameud, O., Hosni, A.I.E. (eds) *Advances in Computing Systems and Applications. Lecture Notes in Networks and Systems,* **1145**. Springer, Cham. (2024) https://doi.org/10.1007/978-3-031-71848-9_31
13. Hornsteiner, M. *et al*. Real-Time Text-to-Cypher Query Generation with Large Language Models for Graph Databases. *Future Internet*, **16**, p. 438. (2024) https://doi.org/10.3390/fi16120438

14. Liang, Y. *et al*. NAT-NL2GQL: A Novel Multi-Agent Framework for Translating Natural Language to Graph Query Language. *arXiv preprint* arXiv:2412.10434. (2024) https://doi.org/10.48550/arXiv.2412.10434
15. Su, X. *et al.* KGARevion: An AI Agent for Knowledge-Intensive Biomedical QA. *arXiv preprint* arXiv:2410.04660. (2024) https://doi.org/10.48550/arXiv.2410.04660
16. Pourreza, M. *et al*. Chase-sql: Multi-path reasoning and preference optimized candidate selection in text-to-sql. *arXiv preprint* arXiv:2410.01943. (2024) https://doi.org/10.48550/arXiv.2410.01943
17. Feng, H., Yin, Y., Reynares, E. and Nanavati, J., OntologyRAG: Better and Faster Biomedical Code Mapping with Retrieval-Augmented Generation (RAG) Leveraging Ontology Knowledge Graphs and Large Language Models. *arXiv preprint* arXiv:2502.18992. (2025) https://doi.org/10.48550/arXiv.2502.18992
18. Zhong, Z. *et al.* Synthet2c: Generating synthetic data for fine-tuning large language models on the text2cypher task. *arXiv preprint* arXiv:2406.10710. (2024) https://doi.org/10.48550/arXiv.2406.10710
19. Soman, K. *et al.* Biomedical knowledge graph-optimized prompt generation for large language models, *Bioinformatics*, **40**, Issue 9, btae560. (2024) https://doi.org/10.1093/bioinformatics/btae560
20. Martin, B.S., Stevens, J.S., Docherty, M., Lee, S. Ask ARCH Graph: LLM Question Answering over a Large-scale Knowledge Graph. *The Knowledge Graph Conference*. Cornell Tech, New York, NY, USA. (2024)
21. Han, H. *et al*. Rag vs. graphrag: A systematic evaluation and key insights. *arXiv preprint* arXiv:2502.11371. (2025) https://doi.org/10.48550/arXiv.2502.11371
22. Edge, D. *et al.* From local to global: A graph rag approach to query-focused summarization. *arXiv preprint* arXiv:2404.16130. (2024) https://doi.org/10.48550/arXiv.2404.16130
23. Wu, J. *et al.* Medical graph rag: Towards safe medical large language model via graph retrieval-augmented generation. *arXiv preprint* arXiv:2408.04187. (2024) https://doi.org/10.48550/arXiv.2408.04187
24. Koscielny, G., *et al*. Open Targets: a platform for therapeutic target identification and validation. *Nucleic Acids Res*. **45**, pp. D985-D994. (2017) https://doi.org/10.1093/nar/gkw1055
25. Lobentanzer, S. *et al.* Democratizing knowledge representation with BioCypher. *Nat Biotechnol* **41**, 1056–1059 (2023). https://doi.org/10.1038/s41587-023-01848-y
26. Garda, S. and Leser, U. BELHD: improving biomedical entity linking with homonym disambiguation. *Bioinformatics*, **40**, p.btae474. (2024) https://doi.org/10.48550/arXiv.2401.05125
27. DSPy: Programming—not prompting—LMs. https://dspy.ai/ (accessed April 21 2025)
28. Opsahl-Ong, K. *et al.* Optimizing instructions and demonstrations for multi-stage language model programs. *arXiv preprint* arXiv:2406.11695. (2024) https://doi.org/10.48550/arXiv.2406.11695
29. Kirchhübel, C., Brown, G. Intellectual property rights at the training, development and generation stages of Large Language Models. In *Proceedings of the Workshop on Legal and*

Ethical Issues in Human Language Technologies @ LREC-COLING. pp 13–18, Torino, Italia. ELRA and ICCL. (2024)

30. Liang, T. *et al.* Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint* arXiv:2305.19118. (2023) https://doi.org/10.48550/arXiv.2305.19118
31. Chen, Q. *et al*. Benchmarking large language models for biomedical natural language processing applications and recommendations. *Nat Commun* 16, 3280 (2025). https://doi.org/10.1038/s41467-025-56989-2
32. Model Context Protocol, https://github.com/modelcontextprotocol (Accessed June 30, 2025)
33. LangChain. https://python.langchain.com/v0.1/docs/modules/agents/agent_types/react/ (Accessed April 21 2025)

Code and Data Availability

The full list of the modified Biomix test questions, additional methodology details, and detailed report of the test results along with the Python code are available in the Supplementary information on Github (https://github.com/PistoiaAlliance/LLM/). Data used for the evaluation are available on request from the corresponding author.

Acknowledgment

This research project was organized by the Pistoia Alliance with financial support from Abbvie and AstraZeneca.

Author contributions

- Vladimir A. Makarov: conceptualization, funding acquisition, supervision, methodology, project administration, writing – original draft, writing – review and editing
- Oleg Stroganov: data curation, formal analysis, software
- Laura I. Furlong: writing – original draft, writing – review and editing
- Brian Evarts: conceptualization, writing – original draft, writing – review and editing
- Loes van den Biggelaar: conceptualization, writing – original draft, writing – review and editing
- Alexandros Goulas: writing – review and editing
- Etzard Stolte: conceptualization, writing – review and editing
- Derek Marren: conceptualization, funding acquisition, methodology, writing – review and editing
- Lars Greiffenberg: conceptualization, funding acquisition, methodology, writing – review and editing

BE, LvdB made equal contributions to this work.

Table 1. Descriptions of strategies used.

Strategy Name	Description
Background knowledge	LLM required to answer the question from its background knowledge, without references to external data sources
Naïve LLM	Naïve generation of Cypher queries. No information about DB schema is provided
Naïve with DB description	Naïve LLM with a DB description in plain text, supplied manually
Naïve with automatically generated schema	Use automatically generated schema in naïve strategy schema was generated with LangChain [33] tools. These tools provided the information about node properties and relationships present in the knowledge graph. Enhanced schema additionally includes examples values per each node and each relationship property
Naïve with GraphQA library	Use Langchain [33] GraphQA library to aid in naïve LLM query
Hard-coded query templates	LLM picks from a few hard-coded templates and substitutes parameter values
Examples of queries	LLM learns from example queries added to the LLM prompt
BioChatter built-in method	Incremental query building strategy in BioChatter, a NL2SL tool built into BioCypher [25]
Agents - single	A single “Data Scientist” agent converses with the user
Agents - multiple	Breakdowns into steps followed by conversation with the user, combined with the multi-agent approach. 2 loops: (1) “Data Scientist” agent with feedback on query execution, (2) QC on results, plus an extra “ask user” tool

Specialized Text2Cypher LLM	Use Text2Cypher [11] of different sizes
Background / Question/Answer with DSPy [27]	Background knowledge of LLM tested using simple question/answering approach (training set for DSPy [27] is available in the Supplemental information in Github)
Background / Chain of Thought with DSPy [27]	Background knowledge of LLM tested using chain of thought prompt
Background / Chain of Thought with DSPy [27] and MIPROv2 [28]	Background knowledge with CoT prompt after prompt optimization with MIPROv2 [28] algorithm
Query from DB schema with DSPy [27]	LLM is asked to construct query from schema and then answer from query results
Query from DB schema with DSPy [27] and MIPROv2 [28]	LLM is asked to construct query from schema and then answer from query results; prompts optimized with MIPROv2 [28] algorithm
ReAct agent framework	ReAct agent based framework, with a single Data Scientist agent and feedback on query results. (No formal QC agent). ReAct is part of the LangChain framework [33].
ReAct agent framework with MIPROv2 [28]	ReAct agent based framework, with a single Data Scientist agent and feedback on query results; prompts optimized with MIPROv2 [28] algorithm
Query only with Chain of Thought	LLM asked to construct query; only query is assessed

Query only with Chain of Thought and MIPROv2 [28]	LLM asked to construct query; only query is assessed; prompt is optimized with MIPROv2 [28] algorithm
Query only with Chain of Thought and MIPROv2 [28] heavy	LLM asked to construct query; only query is assessed; prompt is optimized with MIPROv2 [28] algorithm (heavy optimization settings)

Table 2. Set of three complex test questions based upon the Open Targets database.

Question	What counts as success
Q1: What (or how strong, or is there any) is the evidence between TDP-43 and amyotrophic lateral sclerosis (ALS)	LLM retrieves gene-disease associations between TDP-43 (gene symbol TARDBP) and ALS or amyotrophic lateral sclerosis. Should be ~1400 associations
Q2: What is the evidence linking TDP-43 to cancer in animal models?	LLM uses correct disease names (any of *cancer*, *neoplasm*, *tumor*), correct gene symbol (TARDBP), and uses filter on association type (*AnimalModel*). Should be no associations
Q3: What (or is there) is the clinical evidence linking BRAF to Melanoma?	LLM retrieves associations between BRAF and melanoma. *KnownDrug* association type is explicitly used. Other association types are allowed. Should be at least ~200 associations

Table 3. Percent success on the complex question test set.

Strategy Name	**LLM Type**			
	claude-3-5-sonnet	**gpt-4o**	**o1**	**open-mistral-7b**
Naïve LLM	0	0	0	0
Naïve with DB description	67	17	23	0
Naïve with automatically generated schema	10	0	30	3
Naïve with GraphQA library	17	0	17	0
Hard-coded query templates	100	37	100	0
Examples of queries	87	40	97	3
BioChatter built-in method	20	10	30	0
Agents - single	10	90	-	-
Agents - multiple	77	83	-	-

Research Data

Click here to download Research Data
https://github.com/PistoiaAlliance/LLM/