Supplementary Information:

Natural Language Access Point to Digital Metal-Organic Polyhedra Chemistry in The World Avatar

Simon D. Rihm[1], Dan Tran[2], Aleksandar Kondinski[1], Laura Pascazio[2], Fabio Saluz[1,3], Xinhong Deng[2], Sebastian Mosbach[1,2,4], Jethro Akroyd[1,2,4], Markus Kraft[1,2,4,*]

[1] Department of Chemical Engineering and Biotechnology
University of Cambridge
Philippa Fawcett Drive
Cambridge, CB3 0AS
United Kingdom

[2] CARES
Cambridge Centre for Advanced Research and Education in Singapore
1 Create Way
CREATE Tower, #05-05
Singapore, 138602

[3] D-MAVT
ETH Zurich
Rämistrasse 101
Zurich, CH-8092
Switzerland

[4] CMCL Innovations
No. 9, Journey Campus
Castle Park
Cambridge, CB3 0AX
United Kingdom

* Corresponding author, e-mail address: mk306@cam.ac.uk

Contents

A Retrieval-augmented generation (RAG)

In RAG systems, LLM generation needs not rely exclusively on facts implicitly encoded in model weights, but instead can make use of external data sources [1]. Specifically, the RAG architecture generally comprises two separate components: a *retriever* and a *reader*, also known as *generator*. The retriever gathers relevant facts from external data sources and the reader, which can be any text-to-text model, processes the retrieved data to respond to input questions. As data sources can be updated at a relative low cost and independently from LLM training, RAG systems are capable of incorporating new data in their responses without needing to update LLM weights.

The performance of RAG systems is generally sensitive to retrieval success and the ability of the LLM-reader to process long context windows. Ideally, the retrieved data should contain all the necessary information to answer a given question (high recall) and minimal amount of irrelevant data (high precision). The simplest data store is a collection of unstructured text documents. Despite advances in text search methods such as BM25 [2] and semantic search [3], this type of data repository suffers from relatively low retrieval recall and precision, which force the reader component to fall back on its internal knowledge. Structured data stores such as knowledge graphs and RDBMS databases can enable more targeted retrieval, but they require annotation efforts for data instantiation and more elaborate retrieval methods. With semi-structured data, a mix of retrieval strategies can be leveraged and retrieval performance can be tuned accordingly.

Of relevance to TWA are retrieval methods for knowledge graphs, which fall under two broad categories: semantic parsing and information retrieval. Semantic parsing-based systems convert user queries in natural language to a query language compatible with the underlying data store, such as SPARQL[1], Cypher [4], S-expression [5], KoPL [6], or formulated as a program in a general-purpose language like Python [7, 8]. Meanwhile, information retrieval-based systems extract fragments of knowledge graphs by performing relation classification [9, 10] and vector similarity search [11]. Although the information retrieval approach is generally simpler to implement, it is unable to capture higher-order query operations and often suffers from low recall. Meanwhile, the approach of semantic parsing can capture complex constraints such as numerical comparisons and global-level operations, but the development of an accurate semantic parser is not as straightforward.

Advances in NLP capabilities of LLMs have made the development of semantic parsers easier, rendering semantic parsing the generally preferred approach. Exemplary methods include logical query construction as a multi-step search problem [12] and draft-then-refine, whereby a candidate logical form is first generated via zero-shot [13] or in-context few-shot learning [7, 14, 15] and is then refined to improve alignment to query intent and the knowledge base's schema.

One example of a RAG system for reticular chemistry is MOF Chatbot [16]. However, it operates on the level of documents and requires LLMs that can handle long context windows. It also might not be able to handle multi-hop questions due to the inherent limitations of document-RAG.

[1] https://www.w3.org/TR/sparql11-query/

B In-context learning

A technique commonly employed in this usage of LLMs is **in-context few-shot learning** [17], whereby LLMs are provided with a few examples of text input-output pairs at test time, which help align LLM's behaviour with user expectation. In this section, we provide an overview of in-context learning, including its definition, variants of its setup, and factors influencing its performance.

B.1 Formulation

Few-shot in-context learning, or in-context learning for short, refers to the ability of a model to learn to perform tasks when provided with few demonstrations, also known as examples, and without updating its weights. Formally, given an input text x, text generation model f, instruction I, and demonstration set $D = \{(x_1, y_1), ..., (x_k, y_k)\}$, the model outputs the label y of x as $y = f(I, D, x)$. Although in-context learning was first reported in the GPT-3 model as part of the line of research that experiments with model scaling [17], it has been shown that smaller models can also be trained to perform in-context learning [18].

B.2 Variations

Common variations in in-context learning setups differ in the construction of demonstration set D, output processing, and instruction construction.

A **demonstration set** can be fixed at inference time, generally for simple tasks that can be handled with a small number of demonstrations. Retrieval of a demonstration subset is required for complex tasks that are accompanied by a training set that cannot be reasonably fit into the context window of an LLM. Even for LLMs with long context windows, indiscriminate inclusion of demonstrations in LLM prompts might slow down or introduce noise to inference.

Output processing is any additional treatment applied to LLM output meant to obtain a better result than direct LLM decoding. For example, in a workflow that employs the self-consistency check, an LLM is invoked multiple times to obtain a set of n outputs, of which the majority vote decides the final result [7, 14].

Construction of instruction input into LLMs can be fixed or dynamically adapt to every input query to enhance semantic parsing quality. For instance, knowledge base relations that are the most semantically similar to user questions may be inserted into the LLM prompt to minimise hallucination of non-existent schema elements [7].

C Marie's implementation

Unlike standalone large language model (LLM)-based chatbots that tend to hallucinate scientific facts in low-resource domains, 'Marie' provides fact-oriented responses by augmenting LLM generation with data retrieved from TWA [19, 20]. Furthermore, 'Marie' displays a fine-grained understanding of user intent owing to its semantic parsing component, which can accurately represent logical expressions such as 'boiling point greater than 100°C'—an ability that embedding-based methods found in conventional vector-based retrieval-augmented generation (RAG) systems lack.

C.1 Input rewriter

Our system relies on in-context learning to detect physical quantities broken down into magnitudes, units, and quantity types. Refer to Fig. S1 for the structure of the prompt. Unit conversion is done using the Pint library[2], whereby the target unit is looked up based on the quantity type; if no target unit is registered for a quantity type, the quantity will be converted to the SI base units.

Instruction:
Your task is to detect physical quantities in natural language texts based on the examples given. Please ignore physical quantities with no units and respond with a single JSON object exactly, or 'null' if no physical quantities are present.

Input-output examples:
"Find all chemical species with boiling point above 50°C."

```
{
 "template": "Find all chemical species with boiling point above
 ↪  {}",
 "quantities": [
  {"type": "boiling_point", "value": 50, "unit": "degC"}
 ]
}
```

"What is the solubility of C6H6?"
...

Input:
"Find alcohol solvents with a boiling point between 100°C and 120°C."

Figure S1: *LLM prompt for physical quantity detection.*

[2]https://pint.readthedocs.io/en/stable/

Table S1: *Characteristics of questions found in our semantic parsing dataset.*

Criterion	Variants	Example
Answer set cardinality	single	What is the reference zeolite of framework ABW?
	multiple	Find all steroids with molecular weight around 200 g/mol.
Number of constraints	single	Retrieve all MOPs known to have geometric structure (2-bent)x3(3-pyramidal)x2.
	multiple	Find all polymer lubricants with melting point between 200 K and 300 K.
Hop distance	1	Find transport model of oxygen radical.
	2	What are the boiling points of alkenes?
	3	Show all transport models of species that can be used as fuels, indicate which reaction mechanisms the data are derived from.
Query federation	yes	Compare thermo models of hydrocarbons across all mechanisms they appear in.
	no	Show the optimised geometry of H2 calculated using MP2.

C.2 Semantic parser

The semantic parsing dataset are manually crafted to cater to diverse information needs within the chemical realm of TWA. The examples display varying levels of complexities, such as single- and multi-hop questions, single- and multi-constraint questions. Notably, we include queries that require federation over multiple SPARQL endpoints, such as "Compare thermo models of hydrocarbons across all mechanisms they appear in.". Here, the information of which species are classified as hydrocarbon is located in the `ontospecies` triplestore, while thermo model data are stored in the `ontokin` triplestore. For a full analysis of question characteristics, see Table S1.

Our prompt template contains three slots for the input query, relevant knowledge base relations and semantic parsing demonstrations. Both semantic parsing examples, and knowledge base relations are retrieved on-demand by vector similarity search, with the user question as the search query. Each semantic parsing demonstration (x_i, y_i) is represented by the embedding of the input query, *i.e.* $f_{\text{S-BERT}}(x_i)$, while each relation r is represented by the embedding of its formatted `rdfs:label` and `rdfs:comment` attributes, *i.e.* $f_{\text{S-BERT}} \circ f_{\text{format}}(a_{\text{rdfs:label}}, a_{\text{rdfs:comment}})$. We retrieve $k_{\text{KG\_relations}} = 10$ relations and $k_{\text{demonstrations}} = 10$ demonstrations without tuning these parameters. See Fig. S2 for an example of how our prompt is constructed.

Instruction:
Your task is to translate the input question to an executable data request based on the provided relations and semantic parsing examples. Please respond with a single JSON object exactly.

Relations:

```
{
 "IRI": "os:hasUse",
 "comment": "A relation between a species and its uses or
 ↪  applications."
}
```

...

Input-output examples:
"What are some common usages of aromatic compounds?"

```
{
 "var2cls": { "ChemicalClass": "os:ChemicalClass", "Use": "os:Use"
 ↪  },
 "entity_bindings": { "ChemicalClass": ["aromatic compound"] },
 "triplestore": "ontospecies",
 "query": "SELECT DISTINCT ?ChemicalClass ?Use WHERE { ?Species
 ↪  os:hasChemicalClass/rdfs:subClassOf* ?ChemicalClass . ?Species
 ↪  os:hasUse ?Use . }"
}
```

"What chemicals can be used to regulate pH?" => ...
...

Input:
"Find chemicals commonly used as fuels"

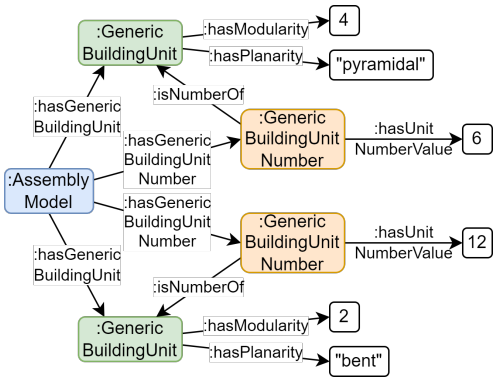
Figure S2: *LLM prompt for semantic parsing.*

C.3 Entity linking

Table S2 summarises strategies for entity linking used in Marie, and Fig. S3 illustrates the entity linking logic for entities of class `AssemblyModel`.

Table S2: *Summary of entity linking strategies and their corresponding illustrative examples. Entities to be linked are in bold.*

	Example	
Strategy	Input question	Entity linking logic
Inverted index lookup	"What is the charge of **benzene**?"	Match against all `rdfs:label`, `skos:altLabel`, IUPAC names, molecular formulae, and SMILES strings of `Species` nodes.
Semantic search	"What chemicals can be used to **regulate pH**?"	Perform semantic search over the labels of all `Use` nodes; entity with label "pH regulator" is matched.
RDF subgraph matching	"Find MOPs with assembly model **(4-pyramidal)x6(2-bent)x12**?"	Execute RDF graph query that matches any `AssemblyModel` entities that are linked to exactly six units of 4-pyramidal and twelve units of 2-bent `GenericBuildingUnit` nodes.



(S3(a))

```
PREFIX : <https://www.theworldavatar.com/kg/ontomops/>

SELECT DISTINCT ?AM WHERE {
  ?AM :hasGenericBuildingUnit ?GBU1;
      :hasGenericBuildingUnitNumber ?GBUNum1.
  ?GBU1 :hasModularity 4; :hasPlanarity
  ↪ "pyramidal".
  ?GBUNum1 :isNumberOf ?GBU1 ;
  ↪ :hasUnitNumberValue 6.

  ?AM :hasGenericBuildingUnit ?GBU2;
      :hasGenericBuildingUnitNumber ?GBUNum2.
  ?GBU2 :hasModularity 2; :hasPlanarity
  ↪ "bent".
  ?GBUNum2 :isNumberOf ?GBU2;
  ↪ :hasUnitNumberValue 12.

  FILTER NOT EXISTS {
    ?AM :hasGenericBuildingUnit ?GBUExclude.
    FILTER ( ?GBUExclude NOT IN  ( ?GBU1 ,
    ↪ ?GBU2 ) )
  }
}
```

(S3(b))

Figure S3: *Illustration of RDF subgraph matching as a strategy for entity linking. (a) The ABox subgraph that defines the assembly model (4-pyramidal)x6(2-bent)x12. (b) the SPARQL query to determine the IRI of this entity.*

D Model verification

In contrast to pre-trained general-purpose LLMs, Marie is limited to the domain knowledge that can be retrieved from the connected knowledge graphs. It performs extremely well within the confines of the domain knowledge model and fails outside of these bounds, which makes comparing traditional benchmarks unnecessary. An in-depth comparison was nonetheless provided for a previous (less capable) version of Marie by Tran et al. [21]. For this specific work, an analysis and rudimentary comparison is shown in Tab. S3.

Table S3: *Analysis of Marie's capability to retrieve MOPs information based on example questions and comparison with augmented ChatGPT.*

Question	**Marie**[a]			**ChatGPT**[b]
	speed[c]	repeatability	accuracy[c]	accuracy
What are all the MOPs with the CBU formula [Mg4C56H76O12S4]?	10.4s	10/10	24.0/24	6/24
Give me all MOPs with a geometric structure of (5-pyramidal)x12(3-planar)x20.	9.1s	10/10	12.0/12	11/12
Return all assembly models based on the CBU [(C6H3)O(CH2)13CH3(CO2)2].	7.3s	10/10	3.0/3	2/3
What assembly models are representative of icosahedral geometry?	5.2s	10/10	2.0/2	1/2
Find the modularity of CBU [(C3N3)(C6H4)3(CO2)3].	6.7s	10/10	1.0/1	0/1
What is the provenance of the information about MOP [V3O2(OH)2(HCO2)3]4 [(C6H4)(C3H2N2)2]6?	7.6s	10/10	1.0/1	0/1
What MOPs are associated with DOI 10.1016/j.chempr.2017.02.002?	6.4s	10/10	1.0/1	0/1
Which chemical building units are used as 2-linear generic building units?	7.0s	10/10	17.0/17	0/17

[a] *Version 6* [b] *Version GPT-4o* [c] *Average values after asking each question 10 times separately.*

Eight representative example questions (not from training set) were asked 10 times each to prove repeatability and answers checked regarding their accuracy and response speed. Speed was measured for the KG information retrieval, generating an additional natural language response takes a bit longer. Average speeds between 5 and 10 seconds are satisfactory for our use case and seems to not only depend on the number of items retrieved but also complexity of the query.

With Marie, the same and correct results were achieved every time. Based on the question, one or more items are retrieved (e.g. 24 MOPs in the first question or a single number in the fifth question). For a fair comparison, ChatGPT (version GPT-4o) was provided with

manuscript and supplementary information files of Kondinski et al. [22]. These contain all information required to answer the questions. As expected, it shows some success but none of the answers are complete and more complex questions could not be answered at all.

References

[1] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474, 2020. doi:10.48550/arXiv.2005.11401.

[2] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009. doi:10.1561/1500000019.

[3] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel Cer, and David Jurgens, editors, *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi:10.18653/v1/S17-2001.

[4] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. Cypher: An evolving query language for property graphs. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD/PODS '18. ACM, May 2018. doi:10.1145/3183713.3190657.

[5] Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. Beyond i.i.d.: Three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, WWW '21, page 3477–3488, 2021. ISBN 9781450383127. doi:10.1145/3442381.3449992.

[6] Shulin Cao, Jiaxin Shi, Liangming Pan, Lunyiu Nie, Yutong Xiang, Lei Hou, Juanzi Li, Bin He, and Hanwang Zhang. KQA pro: A dataset with explicit compositional programs for complex question answering over knowledge base. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6101–6119, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-long.422.

[7] Zhijie Nie, Richong Zhang, Zhongyuan Wang, and Xudong Liu. Code-style in-context learning for knowledge-based question answering. *Proc. AAAI Conf. Artif. Intell*, 38(17):18833–18841, Mar. 2024. doi:10.1609/aaai.v38i17.29848.

[8] Xintao Wang, Qianwen Yang, Yongting Qiu, Jiaqing Liang, Qianyu He, Zhouhong Gu, Yanghua Xiao, and Wei Wang. KnowledGPT: Enhancing large language models with retrieval and storage access on knowledge bases, 2023. URL https://arxiv.org/abs/2308.11761.

[9] Yike Wu, Nan Hu, Guilin Qi, Sheng Bi, Jie Ren, Anhuan Xie, and Wei Song. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. In *Proceedings of The 12th International Joint Conference on Knowledge Graphs*, 2023. doi:10.48550/arXiv.2309.11206.

[10] Jiho Kim, Yeonsu Kwon, Yohan Jo, and Edward Choi. KG-GPT: A general framework for reasoning on knowledge graphs using large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9410–9421, dec 2023. doi:10.18653/v1/2023.findings-emnlp.631.

[11] Rui Yang, Haoran Liu, Edison Marrese-Taylor, Qingcheng Zeng, Yu He Ke, Wanxin Li, Lechao Cheng, Qingyu Chen, James Caverlee, Yutaka Matsuo, and Irene Li. KG-Rank: Enhancing large language models for medical QA with knowledge graphs and ranking techniques, 2024. URL https://arxiv.org/abs/2403.05881.

[12] Yu Gu, Xiang Deng, and Yu Su. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4928–4949, July 2023. doi:10.18653/v1/2023.acl-long.270.

[13] Dean Allemang and Juan Sequeda. Increasing the LLM accuracy for question answering: Ontologies to the rescue!, 2024. URL https://arxiv.org/abs/2405.11706.

[14] Tianle Li, Xueguang Ma, Alex Zhuang, Yu Gu, Yu Su, and Wenhu Chen. Few-shot in-context learning on knowledge base question answering. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6966–6980, July 2023. doi:10.18653/v1/2023.acl-long.385.

[15] Sitao Cheng, Ziyuan Zhuang, Yong Xu, Fangkai Yang, Chaoyun Zhang, Xiaoting Qin, Xiang Huang, Ling Chen, Qingwei Lin, Dongmei Zhang, Saravan Rajmohan, and Qi Zhang. Call me when necessary: LLMs can efficiently and faithfully reason over structured environments, 2024. URL https://arxiv.org/abs/2403.08593.

[16] Yeonghun Kang and Jihan Kim. ChatMOF: an artificial intelligence system for predicting and generating metal-organic frameworks using large language models. *Nat. Commun.*, 15(1):4705, 2024.

[17] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever,

and Dario Amodei. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, 2020. ISBN 9781713829546. doi:10.48550/arXiv.2005.14165.

[18] Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. MetaICL: Learning to learn in context. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809, July 2022. doi:10.18653/v1/2022.naacl-main.201.

[19] Aleksandar Kondinski, Pavlo Rutkevych, Laura Pascazio, Dan N. Tran, Feroz Farazi, Srishti Ganguly, and Markus Kraft. Knowledge graph representation of zeolitic crystalline materials. *Digital Discovery*, 3:2070–2084, 2024. doi:10.1039/D4DD00166D.

[20] Laura Pascazio, Dan Tran, Simon D. Rihm, Jiaru Bai, Sebastian Mosbach, Jethro Akroyd, and Markus Kraft. Question-answering system for combustion kinetics. *Proc. Combust. Inst.*, 40(1):105428, 2024. doi:https://doi.org/10.1016/j.proci.2024.105428.

[21] Dan Tran, Laura Pascazio, Jethro Akroyd, Sebastian Mosbach, and Markus Kraft. Leveraging text-to-text pretrained language models for question answering in chemistry. *ACS Omega*, 9(12):13883–13896, March 2024. doi:10.1021/acsomega.3c08842.

[22] Aleksandar Kondinski, Angiras Menon, Daniel Nurkowski, Feroz Farazi, Sebastian Mosbach, Jethro Akroyd, and Markus Kraft. Automated rational design of metal–organic polyhedra. *J. Am. Chem. Soc.*, 144(26):11713–11728, June 2022. doi:10.1021/jacs.2c03402.