Preprint Cambridge Centre for Computational Chemical Engineering ISSN 1473 – 4273

Leveraging text-to-text pre-trained language models for question answering in chemistry

Dan Tran[1], Laura Pascazio[1], Jethro Akroyd[1,2,3], Sebastian Mosbach[1,2,3], Markus Kraft[1,2,3,4,5]

released: October 23, 2023

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

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

[3] CMCL Innovations
Sheraton House
Cambridge
CB3 0AX
United Kingdom

[4] School of Chemical
and Biomedical Engineering
Nanyang Technological University
62 Nanyang Drive
Singapore, 637459

[5] The Alan Turing Institute
London
United Kingdom

Preprint No. 313

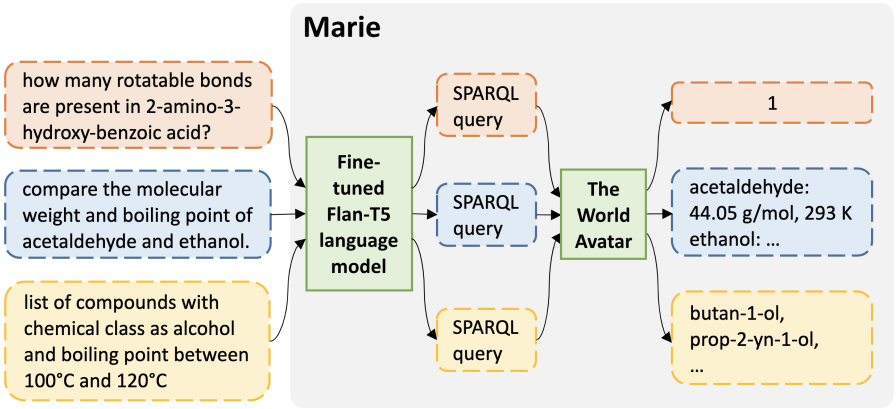
Keywords: text-to-text pre-trained language models, chemical data, KGQA, KG, The World Avatar

Edited by
Computational Modelling Group
Department of Chemical Engineering and Biotechnology
University of Cambridge
Philippa Fawcett Drive
Cambridge, CB3 0AS
United Kingdom

E-Mail: mk306@cam.ac.uk
World Wide Web: https://como.ceb.cam.ac.uk/

Abstract

In this study, we present a question answering (QA) system for chemistry, named Marie, with the use of a text-to-text pre-trained language model to attain accurate data retrieval. The underlying data store is "The World Avatar" (TWA) a general world model consisting of a knowledge graph (KG) that evolves over time. TWA includes information about chemical species such as their chemical and physical properties, applications, and chemical classifications. Building upon our previous work on KGQA for chemistry, this advanced version of Marie leverages a fine-tuned Flan-T5 model to seamlessly translate natural language questions into SPARQL queries, with no separate components for entity and relation linking. The developed QA system demonstrates competence in providing accurate results for complex queries that involve many relation hops, as well as showcasing the ability to balance correctness and speed for real-world usage. This new approach offers significant advantages over the prior implementation that relied on knowledge graph embedding. Specifically, the updated system boasts high accuracy and great flexibility in accommodating changes and evolution of the data stored in the knowledge graph without necessitating retraining. Our evaluation results underscore the efficacy of the improved system, highlighting its superior accuracy and the ability in answering complex questions compared to its predecessor.



Highlights

- A QA system for chemistry that leverages pre-trained language models to translate natural language question in SPARQL queries.
- The QA system can resolve complex queries that involve many relation hops.
- The QA system boasts high accuracy and the flexibility to adapt to changes and evolution in the knowledge graph without necessitating retraining.

Contents

1 Introduction

With the rapid progression of digital technologies, the chemistry sector is generating vast amounts of complex data. Traditional methods for information storage and retrieval are struggling to manage this increasing volume and complexity. The fragmented and often incomplete state of chemical data also poses significant challenges to machine learning. While machine learning has the potential to revolutionize data analysis, its application is inhibited by its need for large quantities of clean data. Much time is often wasted simply gathering and cleaning data. This accentuates the importance of accurate and well-structured data [22, 24].

In this regard, since the landmark publication by Berners-Lee et al. [5], semantic web technologies and Knowledge Graphs (KGs) have been introduced as a solution, providing an effective framework for semantic information retrieval. A KG is a network of data expressed as a directed graph, where the nodes of the graph are concepts or their instances (data items) and the edges of the graph are links between related concepts or instances. KGs are often built using the principles of Linked Data. KGs enable efficient storage and retrieval of interconnected web-scale data and can identify new relationships between various entities. While they may not necessarily speed up queries or reduce hardware requirements, they have the ability to streamline data discovery and cleaning processes that lead to more efficient data utilization and processing. Major existing KGs include the Wikidata KG [54], the DBpedia KG [2], and the Google KG [16]. In the chemistry field, where the need of clean and well-structured data is of fundamental importance, KGs have recently begun to be explored in the context of drug and material discovery and have the potential to assist in key challenges such as target identification [6, 29].

However, querying a KG can be often challenging due to their size and complexity. To submit a query, the user needs to know the formal query language SPARQL [42] and a complete understanding of the KG schema. In some cases of user error, the queries return no data but are considered formally correct, and no warnings are reported.

For this reason, a more user-friendly interface able to retrieve data from KGs is desired. Knowledge Graph Question Answering (KGQA) allows to answer natural language queries posed over the KG. In particular, KGQA in the chemistry domain is a promising area of research owing to the rapid growth of chemistry-related KGs and the potential advantages of a deep search of the chemical space.

Marie [61–63] is a KGQA system for chemistry. It is part of a wider effort to develop a user-friendly interface to interact with The World Avatar (TWA)–a world model built upon a dynamic knowledge graph with the aim to interoperate across heterogenous ontologies and disparate domains, including not just chemistry but also power systems, 3D models of cities and landscapes, among others [33]. Interactions with TWA are facilitated by agents, which are applications and services capable of performing various operations upon the underlying KG. Navigating the suite of agents for varying use cases can be challenging. Ultimately, a natural language interface for TWA could offer a single unified entry point for user interactions by providing a layer of abstraction on top of the many services and data stores available.

The first iteration of Marie [61] was an early attempt at developing a multi-ontology

KGQA system for chemistry and demonstrates the viability of converting questions in natural language to SPARQL queries by matching user's intention to a predefined query template and populate the template with values extracted from user's utterance. This template-based approach [1, 4, 60] is a traditional method falling under the category of semantic parsing, which aims to parse a given question into its logical form, *i.e.* a semantically equivalent representation either in SPARQL or an intermediate form that can be trivially converted to SPARQL [20, 32]. The said system's reliance on hand-crafted templates, which are limited to simple one-hop questions, hampers scalability because the task of expanding the template store falls on domain experts.

More is left to be explored with semantic parsing-based approaches and the use of PLMs. Key approaches to KGQA that have benefited from generative PLMs include end-to-end translation [3, 45, 47], the retriever-reader model [8, 11, 23, 51, 57], and dynamic logical form induction [19, 21]. Even though many of these systems have clinched state-of-the-art results on public benchmarks [3, 21, 51], attempts to quantify their speed to demonstrate practical usage are still inadequate; we are only able to find reports of system speed measured on GPU in two papers [19, 21]. This casts doubt on whether systems with impressive accuracy can be realistically deployed in a real-world setting, where specialized hardware might not be available. More broadly, we notice that efforts to study the accuracy-latency trade-off for KGQA in a hardware-constrained setting are lacking despite the wealth of similar work in other machine learning domains [7, 37, 39, 55, 56].

The purpose of this paper is to present a new version of Marie, a KGQA system that aims to facilitate an extensive exploration of the chemical space. The proposed design of Marie aims to translate questions in natural language to SPARQL queries with the use of a fine-tuned Flan-T5 model [10], known for its demonstrated versatility and efficacy in adapting to downstream tasks [10] while capable of running inference on consumer-grade hardware. Our system adopts a unified pipeline for KGQA, wherein translation is performed end-to-end and without separate components for entity and relation linking, but with simple corrective procedures to counteract the uncertainty in the generation of target queries. We also explore 8-bit quantization as a technique to negotiate the trade-off between accuracy and latency. The developed KGQA system is lightweight and able to not just handle multi-hop questions and but also balance between correctness and speed in a CPU-only setting. This new approach offers significant advantages over the prior implementation that relies on knowledge graph embeddings [63]. Specifically, the updated system boasts higher accuracy and greater flexibility in accommodating changes and evolution of the data stored in the knowledge graph without necessitating retraining.

2 Related work

2.1 Pre-trained Language Models for Knowledge Graph Question Answering

In this section, we provide an overview of major approaches that leverage PLMs for KGQA and where our system stands in this suite of methods.

End-to-end translation. Capitalizing on the generative capabilities of LMs, KGQA systems that adopt the translation approach take in natural language questions and feed them to fine-tuned LMs to output SPARQL queries. For many systems [23, 47], the grounded queries are generated in a one-shot manner. Meanwhile, LMs could be employed to obtain only query skeletons, which are then grounded with entities and/or relations detected by a separate set of components [14, 45]. Recognizing that one-shot generation of executable SPARQL queries is prone to KG misalignment especially for unseen entities and relations, but decoupling entity and relation linking from logical form generation opens up more room for errors, researchers behind the system GETT-QA [3] propose a middle ground, whereby in the first step SPARQL queries are generated with entity and relation slots already filled with their surface forms as found in input questions, and in the second steps these labels are grounded to actual KG entities and relations.

Retriever-reader model. Similar to text-to-text translation, systems that follow the retriever-reader model also utilize fine-tuned LMs to generate SPARQL queries, but they augment the input into the LMs with additional signals. The pipeline of such systems can be broken down into two main steps: in the first step, a retriever processes an input question to gather information relevant to the formation of the corresponding logical form; in the second step, a reader, which is often a fine-tuned LM, takes in the given question and the retrieved information and outputs the desired logical form. Different systems design for different kinds of information to be retrieved and fed into the reader, such as entities and relations detected from input questions [8, 34, 51], candidate logical forms [11], candidate query paths [51, 57], or linearized facts [59]. To ensure that the generated queries are compliant with the ontology of the KG, many of these system impose decoding constraints on the reader [51] or perform an additional step of revision to re-align output logical forms to the KG's ontology [8, 11].

Dynamic logical form induction. Rather than directly generating formal queries in full, systems that perform dynamic logical forms induction starts with a partial query and incrementally expands it till the desired logical form is found, using the discriminative abilities LMs to guide this construction process [19, 21]. The incremental expansion of logical forms enables more fine-grained control over the generation process by limiting the search space of candidate query paths and enforcing grammatical rules.

Relevance to our system. It is critical to assess the applicability of aforementioned approaches with respect to the development of a KGQA system for chemistry. Although dynamic logical form induction has yielded state-of-the-arts results [21], this approach relies strongly on the completeness of the KG to construct logical forms. For instance, for a rare chemical species, it is not uncommon that the data about some of its properties are unavailable. In such a scenario, the dynamic logical form induction approach would fail to generate the appropriate logical form and potentially report its failed prediction, while in fact the expected behaviour would be to produce a valid SPARQL query and return an empty response. End-to-end translation and retriever-reader approaches are both promising, differing only in whether the input into LMs is augmented with extra information other than the question posed to the system. In this work, we adopt the end-to-end translation approach and draw inspiration primarily from GETT-QA [3] due to its straightforward design and leave the exploration of retriever-reader model in future work.

2.2 TWA Chemistry Ontologies

Marie is a KGQA system developed for chemistry, which operates on top of the TWA KG chemistry domain. The TWA KG is a comprehensive, cross-domain, and dynamic knowledge graph adhering to linked data principles. It seamlessly integrates various ontologies, such as OntoSpecies [41], OntoKin [15], OntoCompChem [30], OntoPESScan [36], and OntoMOPs [29], all tailored for representing chemical information. OntoSpecies is a fundamental chemistry ontology within TWA KG. This ontology contains the IRIs of about 36,000 and is constantly growing. The ontology also covers the basic chemical and physical properties of species. It encompass a diverse collection of identifiers, classifications and uses of chemical species, as well as spectral data, in addition to information indicating its origins and attribution. OntoKin is an ontology focused on representing chemical kinetic reaction mechanisms, offering details on reactants and products as well as kinetic, thermodynamic, and transport models [15]. OntoCompChem is an ontology designed for computational chemistry calculations, particularly for density functional theory (DFT) calculations [30]. OntoCompChem currently represents single point calculations, geometry optimizations and frequency calculations. A different ontology, OntoPESScan, has been specifically designed for the representation of potential energy surface (PES) scans [36]. OntoMOPs is an ontology designed for the rational design of metal-organic polyhedra (MOPs) [29]. It encodes assembly models and generic building units as blueprints for creating MOPs, facilitating their design with chemical and spatial reasoning. The current iteration of Marie is specifically configured to operate within the OntoSpecies KG. In this research paper, we aim to showcase the utility of a fine-tuned language model for translating natural language queries into SPARQL queries, particularly for non-shallow ontologies. This approach enables us to address complex questions within the domain effectively. However, our ultimate goal is to broaden the scope of Marie in future iterations to encompass the entire TWA chemistry domain.

3 Methodology

Our system comprises three main steps: pre-processing, translation, and post-processing. In the pre-processing step, special characters present in the input question are encoded and physical quantities are converted to SI units. In the translation step, the pre-processed question is passed to a fine-tuned Flan-T5 model to generate the SPARQL query. In the last step, the predicted query is post-processed to have special characters decoded and additional triple patterns added to enhance user experience. We describe our system architecture in greater details in Section 3.2. Additionally, to train a translation model that learns a mapping from natural language to formal queries, a supervised dataset of question-logical form pairs is needed. Our procedure for constructing this dataset is outlined in Section 3.1.

3.1 Dataset

The scope of our dataset includes information from OntoSpecies KG about chemical species such as their chemical and physical properties, applications, and chemical classifications. Following landmark works in constructing datasets for KGQA [20, 53], we devise a data construction pipeline that runs in three main steps: (1) generate logical forms from a KG, (2) verbalize the logical forms into questions in natural language, and lastly, (3) rephrase the verbalized questions to obtain examples with more linguistic variability. Figure 1 illustrates the data generation process in a nutshell.

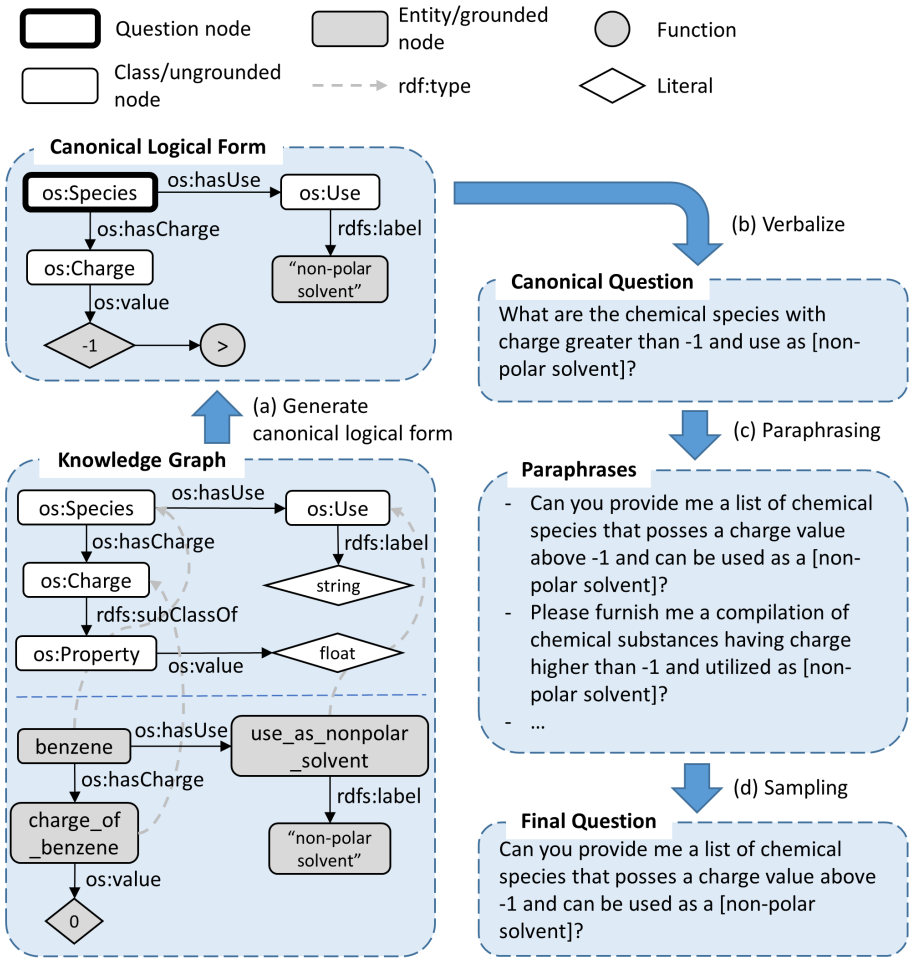


Figure 1: *Steps to generating our dataset. (a) A canonical logical form is generated from a schema subgraph and its instantiation. (b) The logical form is verbalized to form a canonical question. (c) The canonical question is rephrased into different utterances. (d) The final question is sampled from the pool comprising the canonical question and its valid paraphrases.*

3.1.1 Canonical logical form generation

Emulating the methods used in the creation of general-domain KGQA datasets GrailQA [52] and GraphQuestions [20], we traverse the KG's ontology to obtain ungrounded subgraphs consisting of classes and their relations; these subgraphs subsequently have some of their nodes grounded and functions added, yielding canonical logical forms. We control the level of complexity of our generated queries by applying some heuristics on the structure of the retrieved subgraphs and the choice of grounded and question nodes. The out-degree of the `os:Species` is chosen to vary from 1 to 3, and the grounded and question nodes are chosen such that we obtain three types of queries: those that retrieve data about a specified chemical species or more, those that search for chemical species that satisfy a set of conditions constrained upon their properties, and those that inquire about the properties of chemical species belonging a given chemical class. See Figure 2 for examples of these query types.

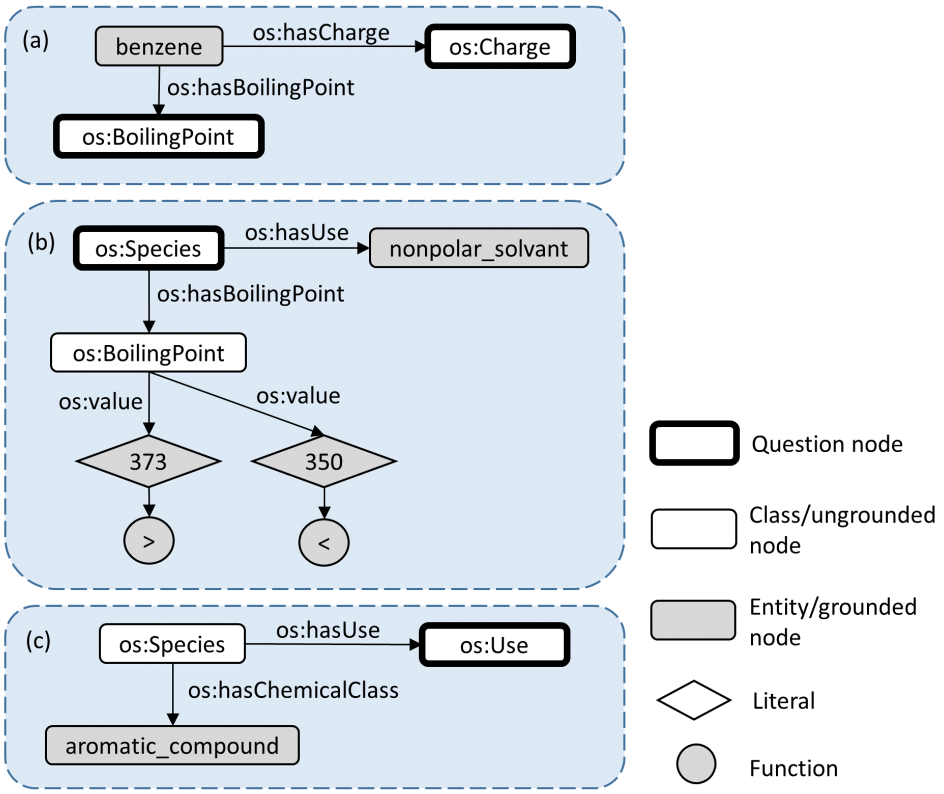


Figure 2: *An overview of query types included in our dataset. (a) One-hop queries that look up information about a given chemical species or more e.g. "What are the boiling point and charge of benzene?". (b) One-hop queries that find chemical species based on specified criteria e.g. "what are the chemical species that can be used as a nonpolar solvant and have a boiling point between 373 and 350?". (c) Two-hop queries that ask about properties of chemical species belonging to a particular chemical class e.g. "what are the applications of chemical species classified as aromatic compound?".*

In contrast to the aforementioned general-domain KGQA datasets that include up to only one function that can be a counting operation, a superlative (arg max, arg min), or a comparative ($>$, $\geq$, $<$, $\leq$) [20, 52], we apply one comparative operator out of the five choices shown in Table 1 on all numerical qualifiers. The rationale for this is to accommodate practical scenarios of data lookup in the chemistry domain, such as the use case of finding solvents whose boiling point falls within a certain range when performing distillation.

Table 1: *Comparative operators present in our dataset.*

	Logical form	**Verbalization**
higher	$x > a$	higher than a
lower	$x < a$	lower than a
inside	$a < x < b$	inside the range between a and b
outside	$x < a \parallel x > b$	outside the range between a and b
around	$0.9a < x < 1.1a$ for $a > 0$	around a

Unlike most KGQA datasets, we do not include entity IRIs in our generated SPARQL queries, and instead perform entity linking directly within the SPARQL queries by exact string matching with node labels; for the case of entities of the class `os:Species`, linking is done by exact matching with any of its chemical identifiers of any subclass of `os:Identifier`.

3.1.2 Verbalization

To support the different ways that a query intent can be formulated and submitted to QA systems, we consider three kinds of verbalizations: the interrogative form, the imperative form, and keyword search. Each of these yields a different canonical question; see Table 2 for an example.

Table 2: *An overview of verbalization types used for the formulation of canonical questions.*

Verbalization type	**Example**
Interrogative form	"What is the charge of benzene?"
Imperative form	"Tell me about the charge of benzene."
Keyword search	"charge of benzene"

3.1.3 Paraphrasing

Paraphrasing serves the purpose of capturing different sentence structures that express the same query intent, as well as the various surface forms that an entity or relation can appear in. For example, the two questions "What is the boiling point of water?" and "At

what temperature does water boil at?" correspond to the same SPARQL query involving a single hop over the relation `os:hasBoilingPoint`.

With the assumption that entity linking is done by exact string matching of detected entity spans, we do not alter entity mentions in query verbalizations during paraphrasing; in other words, only the surface forms of relations are rephrased. We perform paraphrasing to only verbalizations in the interrogative and imperative forms, using OpenAI's chat completion API with the gpt-3.5-turbo model. For each canonical question, 5 paraphrases are generated, which are manually checked for semantic correctness; paraphrases that deviate from the original meaning are rejected. Lastly, the final question is sampled from a pool comprising the canonical question and its valid paraphrases.

3.1.4 Dataset analysis

Three datasets are generated: (1) the train set, which is used for fine-tuning of pre-trained language models; (2) the dev set, which helps with model selection during the fine-tuning process; and (3) the test set, which enables unbiased evaluation of fine-tuned models. In total, our dataset covers 56 relations defined in the ontology of OntoSpecies. See Table 3 for the statistics on some characteristics of our dataset.

Table 3: *Statistics on some characteristics of our dataset.*

	size	**% of examples with x relations**			**% of function occurrence**				
		$x = 1$	2	3	higher	lower	inside	outside	around
train	1608	57.84	37.25	4.91	8.27	8.77	7.59	8.58	9.08
dev	180	65.56	32.22	2.22	11.11	6.11	8.33	8.89	7.78
test	182	63.73	33.52	2.75	10.44	9.34	7.69	8.79	7.14

3.2 System architecture

Flan-T5 [10], which is the improved version of its predecessor T5 [44], is our model of choice. The T5 model family has been mobilized in several KGQA systems [3, 8, 34, 48, 51, 57, 59] owing to its open-source nature and the ability to run on consumer-grade hardware.

One issue with T5 is that it is trained specifically on natural language texts, thus the T5 tokenizer treats characters such as the less-than sign ("<") and the curly braces ("{" and "}") as unknown tokens. To handle this out-of-vocabulary problem, we map these characters into alternative string representations, specifically their corresponding HTML entities ("<", "{", "}"). For symmetry, we also convert the greater-than sign to its HTML entity counterpart (">" is substituted with ">"). Below are outlined the additional preprocessing and postprocessing operations employed by our system. See Figure 3 for a running example of a query passing through our KGQA system.

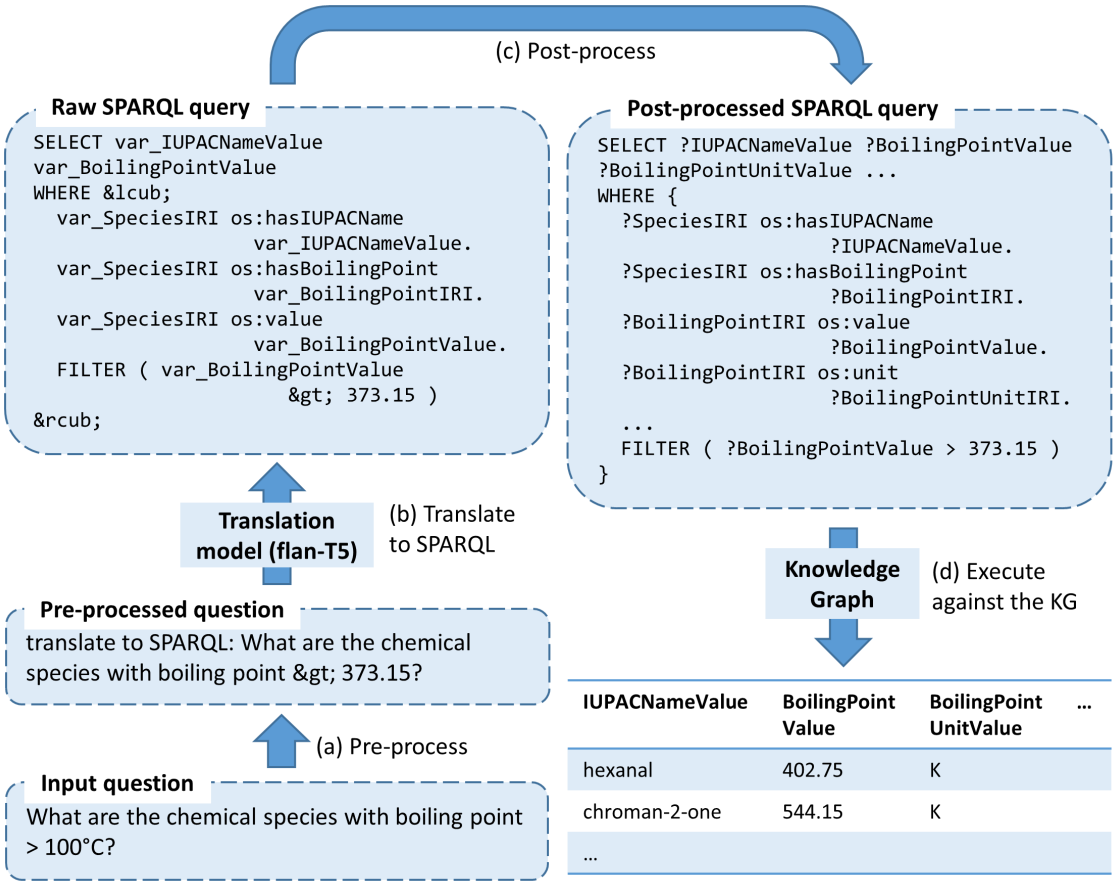


Figure 3: *The key steps in our KGQA system. (a) Physical quantities in the input question are converted to SI units, special characters are encoded, and an instruction prompt is prepended. (b) The question is fed into a fine-tuned Flan-T5 model to obtain SPARQL translation. (c) Special tokens in the translated query are decoded and additional node patterns are added to enhance user experience. (d) The query is executed against the KG to obtain the results.*

3.2.1 Pre-processing of input texts

Physical quantities are only meaningful so long as their units are specified, and different users under varying scenarios might have their own preferred unit system that they find more convenient to work with. To facilitate interoperability with different unit systems, we convert any mentions of physical quantities in the input questions to SI units, which is the unit system used by the OntoSpecies knowledge graph. Here, we assume that units indicated by the user are always valid for the invoked physical quantities e.g. when talking about temperatures the user would use degree Celsius, degree Fahrenheit or Kelvin and not kilogram.

Following unit conversion, characters that are out of T5's vocabulary are encoded into HTML entities as aforementioned. Lastly, to keep in line with the training and fine-tuning approaches of Flan-T5 [10, 44], we prepend the input question with the instruction prompt

"translate to SPARQL: ".

3.2.2 SPARQL encoding

SPARQL encoding determines the representation of SPARQL queries that the language model learns. Besides encoding out-of-vocabulary characters, we follow [58] in converting the prefix marker of query variables, which is the question mark character `?`, to the string prefix `var_`. As mentioned earlier, the design of T5 training and tokenization and its intended application is for natural language. With the assumption that the syntactical function of a question mark is to delimit the end of an interrogative statement, the T5 tokenizer by default strips all whitespaces that appear before a question mark. When this behaviour is applied to SPARQL, a triple pattern such as `"a ?b c"` is read by the model as `"a?b c"`, rendering the role of the question mark ambiguous: is it a prefix, a suffix, or a delimiter? To clearly designate the SPARQL syntax for query variables, we instead use a string prefix. See Table 4 for an illustration.

Table 4: *An example of how a SPARQL query is encoded in our KGQA system. The substitutions are in bold.*

Original SPARQL query

```
SELECT ?SpeciesIRI ?BoilingPointValue
WHERE {
  ?SpeciesIRI os:hasBoilingPoint ?BoilingPointIRI .
  ?BoilingPointIRI os:value ?BoilingPointValue .
  FILTER ( ?BoilingPointValue > 373 )
}
```

Encoded SPARQL query

```
SELECT var_SpeciesIRI var_BoilingPointValue
WHERE &lcub;
  var_SpeciesIRI os:hasCharge var_BoilingPointIRI .
  var_BoilingPointIRI os:value var_BoilingPointValue .
  FILTER ( var_BoilingPointValue > 373 )
&rcub;
```

3.2.3 Post-processing of predicted SPARQL queries

SPARQL queries generated by the language model are in the encoded representation as specified in 3.2.2. Therefore, they are first decoded to recover the original form. The subsequent post-processing procedures are as follows.

Triple pattern expansion. In theory, the retrieval of entity nodes as answers is sufficient for question answering. However, users might require additional information for the answers to be meaningful and human-readable. For instance, when inquiring about the boiling point of a chemical species, it is important to also display the reference state at which the boiling point is measured. Therefore, after decoding SPARQL queries predicted

by our translation model, we augment them with additional triple patterns depending on the class of the answer nodes. See Table 5 for an example.

Table 5: *An example of triple pattern expansion in our post-processing step. The added query variables and triple patterns are in bold.*

SPARQL query with minimal triple patterns

```
SELECT ?SpeciesIRI ?BoilingPointValue
WHERE {
  ?SpeciesIRI os:hasBoilingPoint ?BoilingPointIRI .
  ?BoilingPointIRI os:value ?BoilingPointValue .
  FILTER ( ?BoilingPointValue > 373 )
}
```

SPARQL query with expanded triple patterns

```
SELECT ?SpeciesIRI ?IUPACNameValue ?BoilingPointValue
       ?UnitValue ?RefStateValue ?RefStateUnitValue
WHERE {
  ?SpeciesIRI os:hasIUPACName ?IUPACNameIRI .
  ?IUPACNameIRI os:value ?IUPACNameValue .
  ?SpeciesIRI os:hasBoilingPoint ?BoilingPointIRI .
  ?BoilingPointIRI os:value ?BoilingPointValue ;
                   os:unit ?UnitIRI .
  ?UnitIRI rdfs:label ?UnitValue .
  OPTIONAL {
    ?BoilingPointIRI os:hasReferenceState ?RefStateIRI .
    ?RefStateIRI os:value ?RefStateValue ;
                 os:unit ?RefStateUnitIRI .
    ?RefStateUnitIRI rdfs:label ?RefStateUnitValue .
  }
  FILTER ( ?BoilingPointValue > 373 )
}
```

Copy correction. Our preliminary experiments show that PLMs face difficulties copying exact text spans of chemical species when the surface forms are long, repetitive patterns, as illustrated by Table 6. While researchers have come up with elaborate copy mechanisms that often involve reworking network architectures [9, 18, 23, 49], we make no alteration to the underlying translation model. Instead, we assume that the model is able to detect correct text spans but might not be able to generate them with absolute copying fidelity. To rectify the copy error, we match the model's generated text spans with the closest substrings of the input question, using the Levenshtein distance as the distance metric.

Relation correction. We follow [3] in realigning predicted relations to the actual ones in the ontology of OntoSpecies using an embedding matching mechanism. We employ the Sentence-BERT [46] model, which has been trained using a Siamese network for the semantic matching task, to map relations to low-dimensional vector representations and match them using the cosine similarity as the distance measure.

Table 6: *An example of a question about a chemical species represented by its SMILES string. The corresponding SPARQL query has to copy the string span exactly.*

Question	Share information regarding the optical rotation of
	`CC1C(C(CC(O1)OC2C(OC(CC2O)OC3C(OC(CC3O)`
	`OC4CCC5(C(C4)CCC6C5CCC7(C6(CCC7C8=CC(=O)`
	`OC8)O)C)C)C)C)O)O`

SPARQL query	`SELECT ?OpticalRotationValue`
	`WHERE {`
	`  VALUES ( ?species ) {`
	`  ( "CC1C(C(CC(O1)OC2C(OC(CC2O)OC3C(OC(CC3O)`
	`  OC4CCC5(C(C4)CCC6C5CCC7(C6(CCC7C8=CC(=O)`
	`  OC8)O)C)C)C)C)O)O" ) }`
	`  ?SpeciesIRI ?hasIdentifier ?IdentifierIRI .`
	`  ?IdentifierIRI os:value ?species .`
	`  ...`
	`}`

3.3 Fine-tuning

For fine-tuning, we update all model parameters. We use the AdamW optimizer [35] with a learning rate of 2×10^{-4} and $\varepsilon = 1 \times 10^{-6}$. We keep a constant batch size of 32 across experiment runs and adjust the number of gradient accumulation steps as needed. We train for a fixed number of 3 epochs and perform no hyper-parameter tuning. All fine-tuning is done in a distributed, data-parallel setting, under a hardware budget of a single node consisting of 4x NVIDIA A100-SXM-80GB GPUs.

3.4 Evaluation

Translation quality. The translation quality of our system is evaluated using two automated metrics: the SacreBLEU score [43] and accuracy. SacreBLEU is a variant of the popular BLEU score used for the evaluation of machine translation systems by comparing token-level similarity between reference texts and candidate texts; SacreBLEU has been preferred over BLEU for the former's reproducibility and ease of comparison. The metric runs on a scale from 0 to 100, with higher values indicating higher degrees of lexical match. For translation accuracy, we assign a score of 1 if the machine-translated output is an exact word-level match with the gold reference.

Quantization. Quantization is the technique that maps floating point values to a quantization space supported by fewer bits. This is an established method to significantly reduce inference latency and memory requirement of neural networks with manageable impacts on accuracy [12, 13, 17, 25, 27, 50]. We employ 8-bit dynamic quantization, whereby the weights of neural networks are quantized to 8-bit integers before test time and activations are dynamically quantized during inference. We evaluate the performance of our system under two settings of floating point precision: full precision (32-bit) and 8-bit

quantization.

Hardware. Fine-tuned models are evaluated in a CPU-only setting with a 12th Gen Intel(R) Core(TM) i7-1270P 2.20 GHz processor and 64GB RAM. To mimic a production environment, inference is run with a more performant model format and interpreter, which are chosen to be the ONNX format[1] and and ONNX Runtime[2].

4 Results and discussion

4.1 Quantitative results and error analysis

Table 7 shows the quantitative results for the translation quality of our KGQA system with different base models and post-processing procedures. These results indicate that larger models are able to attain higher accuracies and benefit less from manually-crafted post-processing steps. The fined-tuned Flan-T5-XXL model, which has three billion parameters, attains a near-perfect translation accuracy score of 98.90%.

Table 7: *Translation qualities for various fine-tuned models. The number of parameters of base models are given in brackets next to their names.*

Model	**SacreBLEU**	**Δ**	**Accuracy (%)**	**Δ**
Flan-T5-Small (60M)	78.13	-	36.26	-
+ copy correction	77.86	-0.27	37.36	+1.10
+ relation correction	79.79	+1.93	43.96	+6.60
Flan-T5-Base (250M)	95.43	-	63.19	-
+ copy correction	95.46	+0.03	63.74	+0.55
+ relation correction	96.12	+0.66	68.13	+4.39
Flan-T5-Large (780M)	98.75	-	92.31	-
+ copy correction	98.75	+0.00	92.31	+0.00
+ relation correction	99.21	+0.46	95.60	+3.29
Flan-T5-XL (3B)	99.56	-	95.60	-
+ copy correction	99.56	+0.00	97.25	+1.65
+ relation correction	**99.69**	+0.13	**98.90**	+1.65

To better understand that capacity of PLMs to learn mappings between natural language questions and SPARQL queries in the chemistry domain and the effect of scaling the model size, we conduct an error analysis that classifies incorrect predictions by aspects of logical forms, as summarized in Table 8. We observe that while a small model such as the Flan-T5-Small variant, which has only 60 million parameters, is able to learn the surface representations of SPARQL to some extents, as seen in the unadjusted SacreBLEU score of 78.13, it fails to acquire the implicit syntactical rules of SPARQL, resulting in 10.99%

[1] https://onnx.ai/

[2] https://onnxruntime.ai/

of predictions with incorrect SPARQL syntax; meanwhile, this figure drops to zero for the Flan-T5-XL variant. Smaller models are also less able to handle the diverse query structures that our dataset encompasses and perform poorer at relation prediction. Of note is that smaller models such as Flan-T5-Small and Flan-T5-Base face difficulties learning the mappings for comparative operators, especially when disambiguating the 'inside' and 'outside' functions.

Table 8: *Percentages of incorrect predictions classified by aspects of logical forms.*

Model	% of incorrect predictions			
	syntax	query structure	relation	function
Flan-T5-Small	10.99	24.18	7.69	17.03
Flan-T5-Base	2.75	7.14	4.40	15.93
Flan-T5-Large	1.10	1.65	2.20	0
Flan-T5-XL	0	0.55	0.55	0

4.2 The accuracy-latency trade-off

Next, we quantify the translation latency for varying model sizes and quantization settings. As expected, larger models, although better at natural language-to-SPARQL translation, require more resources to run inference. As can be seen in Table 9, while larger networks produce more accurate results—about 25% increment in accuracy for every increase in the scale of the base model—their translation latency is slower by 2.5 to 3 times, and their memory consumption rises by 1.5 to 2.5 times.

Table 9: *Trade-offs between translation accuracy, translation latency, and memory consumption with varying neural network sizes and quantization treatment, in a CPU-only setting. As translation latency varies with query complexity, see Figure 7. for its distribution.*

Model	Accuracy (%)	Δ	Translation latency (s)	Speedup	Memory (GiB)	%Δ
Flan-T5-Small	43.96	-	1.15	-	3.18	-
+ quantization	21.98	-21.98	0.79	x1.46	2.76	-13.21
Flan-T5-Base	68.13	-	3.00	-	4.32	-
+ quantization	66.48	-1.65	1.72	x1.74	3.29	-23.84
Flan-T5-Large	95.60	-	8.24	-	7.70	-
+ quantization	96.70	+1.10	4.05	x2.03	4.41	-42.73
Flan-T5-XL	98.90	-	25.15	-	19.21	-
+ quantization	89.91	-10.99	9.48	x2.65	8.55	-55.49

Experiments with quantized models at test time reveal that quantization can significantly reduce inference time and memory requirement while maintaining similar levels of accuracy for medium-sized neural networks. Particularly, quantized Flan-T5-Base and Flan-T5-Large models experience slight deviations to their accuracy within a 2% margin, yet their translation latency is sped up by up to 2 times and their memory consumption drops by up to 40%. Furthermore, the narrow increase in accuracy observed for the Flan-T5-Large variant indicates that quantization could have a regularization effect, thus enabling the quantized model to perform better than the full-precision baseline, similarly to previous work in literature [26, 28, 31]. In contrast, quantization of the Flan-T5-XL variant incurs a 10.99% drop in accuracy. This anomalous deterioration is a recognized phenomenon that emerges in LMs exceeding a certain size [12]. While there exists quantization schemes to rectify this anomalous behaviour [12], the ecosystem is still in its nascent stage and the implementation for our specific model architecture and inference runtime is yet developed and rigorously tested. We therefore do not attempt to explore the use of specialized quantization procedure in our system.

Our systematic quantification of accuracy and translation latency helps us establish the Pareto front for the multi-objective optimization of accuracy and latency of our KGQA system, as depicted in Figure 4. This could prove instructive to the practical deployment of a KGQA system. If the Euclidean distance from the ideal point corresponding to 0 latency and 100% accuracy were to be taken as the objective function, i.e. a quadratic objective function, our data support the claim that the quantized Flan-T5-Large model provides the optimal compromise between accuracy and speed, striking an amount of CPU memory consumption of less than 5 GiB and an accuracy score of 96.70%.

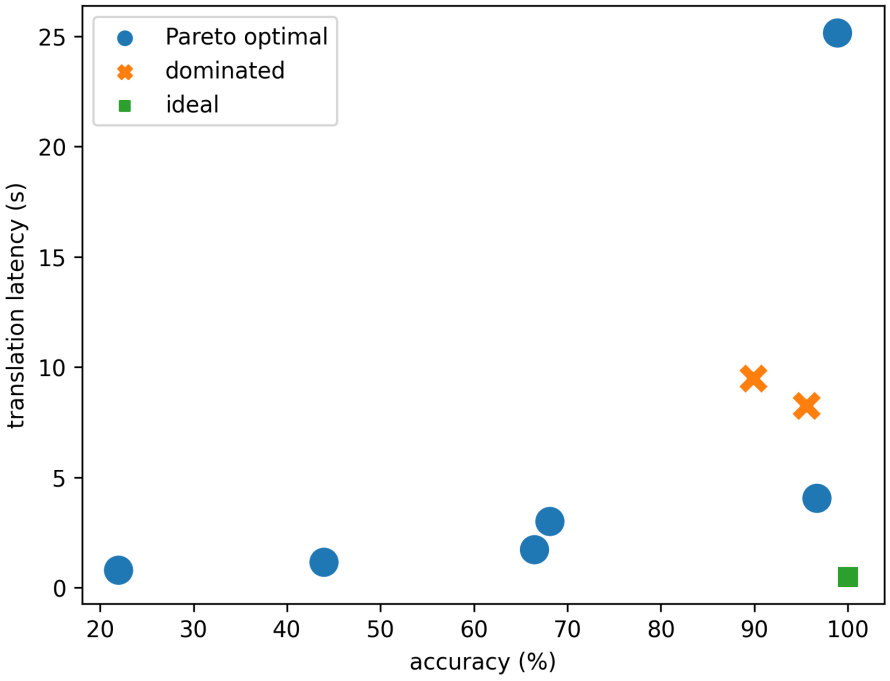


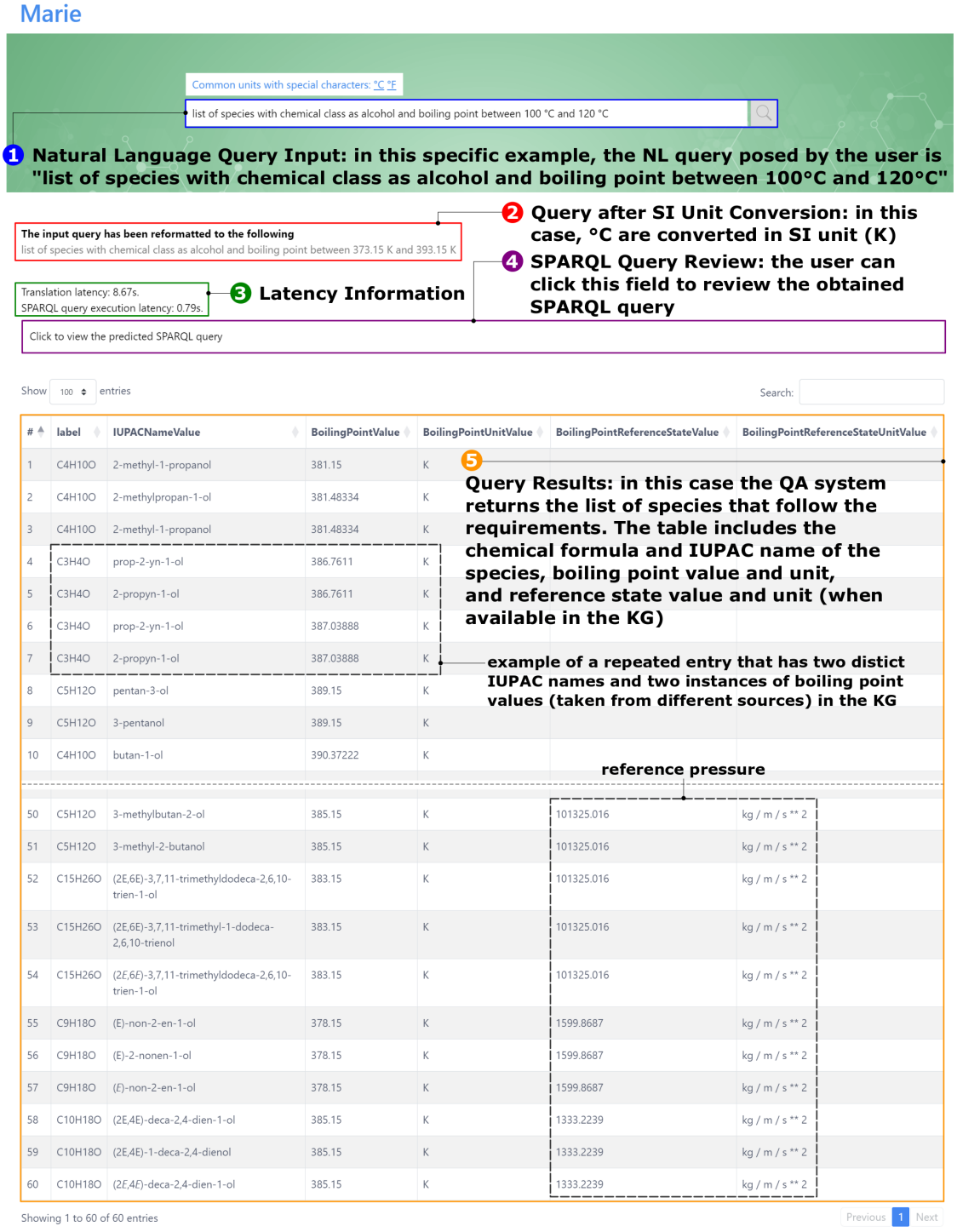
Figure 4: *A scatterplot showing the accuracy-latency trade-off of our KGQA system under a CPU-only setting.*

4.3 Example use case and QA systems comparison

OntoSpecies KG was designed to permit complex queries and easy data analysis and processing in the general chemistry domain. The information reported on chemical species can be used to compare chemical properties of similar compounds, find compounds with required characteristics as well as automate laborious data gathering from research activities [41]. In this section we showcase how Marie can be used for some of these tasks without the requirement of SPARQL knowledge and KG schema knowledge.

Take, for instance, the challenge of identifying property trends for specific chemical categories, like the boiling points of species identified as alcohols. Traditionally, this would involve first identifying the species tagged as alcohols and then delving into their associated properties—a process that can be time-consuming when relying on conventional online sources. However, with the integration of KGs and Marie, this task becomes straightforward by simply asking our QA system. An example can be seen in Fig. 5 where a screenshot of Marie interface showcases a query made in natural language: "list of compounds with chemical class as alcohol and boiling point between 100°C and 120°C". Upon entering the question into the query box (as indicated by field (1) in Fig. 5), the search engine refines the question to match the SI unit standards. This revised question is then displayed in field (2). The engine subsequently produces a table containing the chemical formula, IUPAC name, boiling point values and units, and, when available in the KG, reference pressure values and units (field (5) in Fig. 5). This outcome is akin to what it is obtained from a direct SPARQL query on the KG. Users have the option to review the generated SPARQL query by selecting field (4). Additionally, the time taken for translation and the execution latency of the SPARQL query are displayed in field (3). It's important to mention that each entry in the table is displayed at least twice due to every compound in the KG having two distinct IUPAC names. In some cases, the redundancy extends beyond this because of the manner in which OntoSpecies KG sources its data from PubChem [38, 41]. Given that PubChem aggregates properties from a variety of sources, it's not uncommon to encounter multiple instances of the same property. A case in point is 1-propyn-1-ol, which is listed in Fig. 5 with two separate boiling point values, each attributable to a distinct source. As a result, the system fetches every possible combination of the queried data. Nevertheless, in upcoming versions of Marie, we plan to implement a filter in the SPARQL query to prevent the display of duplicate entries.

Figure 6 presents a comparative view of responses from Marie, Marie and BERT [63] and ChatGPT-4 [40] when posed with the same question. In contrast to our updated version, the earlier Marie and BERT couldn't provide an answer. This limitation stems from the multi-hop queries, which our prior version couldn't handle. Another shortcoming of Marie and BERT is its inability to manage units effectively (questions must be framed using SI units), and it also lacks details regarding the reference state. When posing a question to ChatGPT-4, the system initially prompts the user to reference a database or upload a specific PDF source. If we indicate that we don't have a particular source and request information based on its inherent knowledge, ChatGPT-4 produces a concise table featuring four species. Notably, two of these species are marginally outside the specified range but are included for completeness. The remaining two (1-butanol and 2-methylpropan-1-ol) are also returned by Marie (species highlighted in yellow in Fig. 6). While ChatGPT-4's response is more descriptive, correctly uses units, and provides data on the reference state,



#	label	IUPACNameValue	BoilingPointValue	BoilingPointUnitValue	BoilingPointReferenceStateValue	BoilingPointReferenceStateUnitValue
1	C4H10O	2-methyl-1-propanol	381.15	K		
2	C4H10O	2-methylpropan-1-ol	381.48334	K		
3	C4H10O	2-methyl-1-propanol	381.48334	K		
4	C3H4O	prop-2-yn-1-ol	386.7611	K		
5	C3H4O	2-propyn-1-ol	386.7611	K		
6	C3H4O	prop-2-yn-1-ol	387.03888	K		
7	C3H4O	2-propyn-1-ol	387.03888	K		
8	C5H12O	pentan-3-ol	389.15	K		
9	C5H12O	3-pentanol	389.15	K		
10	C4H10O	butan-1-ol	390.37222	K		
50	C5H12O	3-methylbutan-2-ol	385.15	K	101325.016	kg / m / s ** 2
51	C5H12O	3-methyl-2-butanol	385.15	K	101325.016	kg / m / s ** 2
52	C15H26O	(2E,6E)-3,7,11-trimethyldodeca-2,6,10-trien-1-ol	383.15	K	101325.016	kg / m / s ** 2
53	C15H26O	(2E,6E)-3,7,11-trimethyl-1-dodeca-2,6,10-trienol	383.15	K	101325.016	kg / m / s ** 2
54	C15H26O	(2*E*,6*E*)-3,7,11-trimethyldodeca-2,6,10-trien-1-ol	383.15	K	101325.016	kg / m / s ** 2
55	C9H18O	(E)-non-2-en-1-ol	378.15	K	1599.8687	kg / m / s ** 2
56	C9H18O	(E)-2-nonen-1-ol	378.15	K	1599.8687	kg / m / s ** 2
57	C9H18O	(*E*)-non-2-en-1-ol	378.15	K	1599.8687	kg / m / s ** 2
58	C10H18O	(2E,4E)-deca-2,4-dien-1-ol	385.15	K	1333.2239	kg / m / s ** 2
59	C10H18O	(2E,4E)-1-deca-2,4-dienol	385.15	K	1333.2239	kg / m / s ** 2
60	C10H18O	(2*E*,4*E*)-deca-2,4-dien-1-ol	385.15	K	1333.2239	kg / m / s ** 2

Figure 5: *Illustration of Marie's interface for the query example "list of compounds with chemical class as alcohol and boiling point between 100°C and 120°C". The figure highlights (1) natural language query input, (2) query after SI unit conversion, (3) latency information, (4) SPARQL query review, and (5) query results.*

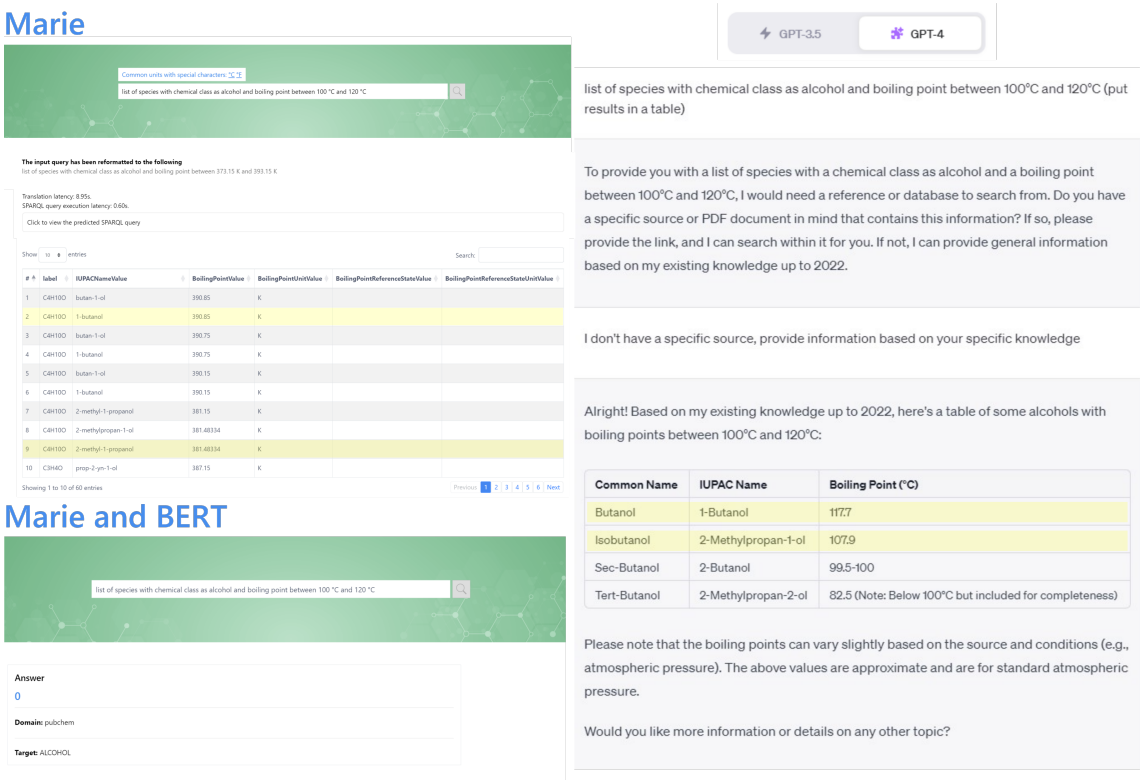


Figure 6: *Comparative visualization of responses from Marie (top left), Marie and BERT (bottom left), and ChatGPT-4 (right) to the query "list of compounds with chemical class as alcohol and boiling point between 100°C and 120°C".*

it isn't as comprehensive as one might hope. This highlights the importance of integrating external databases (in our case OntoSpecies KG) in the QA system to achieve more exhaustive and precise results. Our system excels in delivering precise results and seamlessly adapts to the evolving nature of the database. As the KG expands, the responses generated become increasingly comprehensive and accurate. This dynamic adaptability is a significant advantage, as it eliminates the need for periodic retraining. This means that as more data is added or updated in the KG, our system can instantly leverage this new information to provide richer and more informed answers, ensuring users always have access to the most current and accurate data available.

Users can interact with Marie by following this link (`https://theworldavatar.io/chemistry/documentation/marie`). However, the system is still under development, and the accuracy of the results will increase with further refinement of the underlying ontologies.

5 Conclusions

In this paper, we develop a KGQA system for the chemistry domain that performs end-to-end translation of natural language questions to SPARQL queries, with no separate com-

ponents for entity or relation linking. Additionally, we conduct a hardware-constrained search to find a lightweight configuration for our system that offers a satisfactory compromise between accuracy and speed in a CPU-only setting.

In future work, we will expand the system to work with more ontologies to facilitate a wider range of cross-domain use cases in chemistry-related research and industrial applications.

Data and Software Availability

The code for data generation, fine-tuning, evaluation and deployment is available in the GitHub repository https://github.com/cambridge-cares/TheWorldAvatar/tree/main/MARIE_SEQ2SEQ. All third-party software used in this system is free and available.

Acknowledgements

This research was supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme. Part of this work was supported by Towards Turing 2.0 under the EPSRC Grant EP/W037211/1 and The Alan Turing Institute. M. Kraft gratefully acknowledges the support of the Alexander von Humboldt Foundation. For the purpose of open access, the author has applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising.

A Appendix

A.1 Supplementary figures

Due to the cross-attention mechanism for generating next tokens, the runtime of the Flan-T5 decoder grows quadratic with the output sequence. As the SPARQL queries in the test set vary in length depending on their complexity, the decoding time is expected to display considerable deviations from the mean. Therefore, in addition to the mean values indicated in Table 9, we also report the distribution of the translation latency in Figure 7.

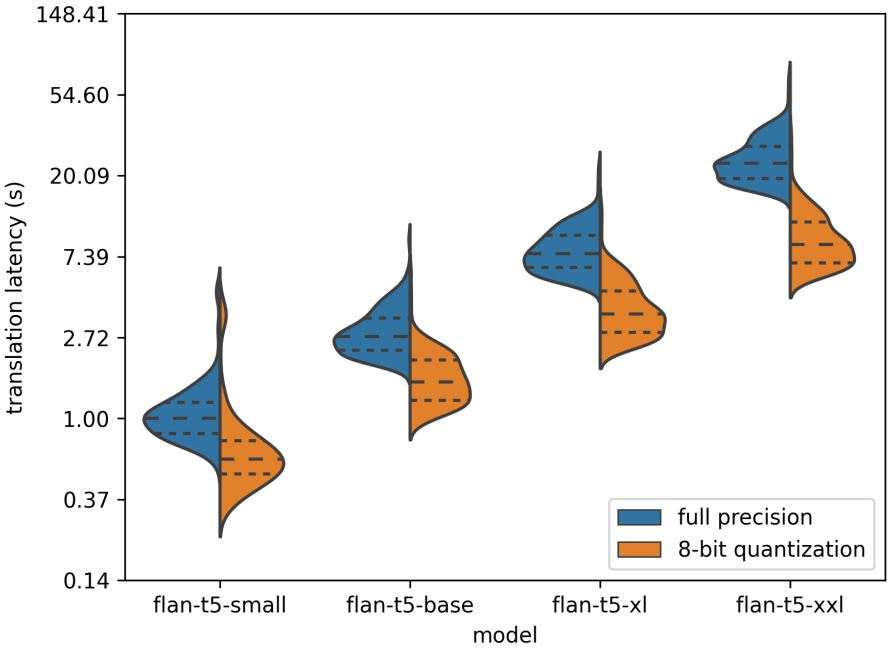


Figure 7: *Distribution of translation latency with varying base models and quantization settings. Note that the y-axis is in the log scale.*

References

[1] A. Abujabal, M. Yahya, M. Riedewald, and G. Weikum. Automated template generation for question answering over knowledge graphs. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 1191–1200, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee. doi:10.1145/3038912.3052583.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. pages 722–735. Springer, 2007.

[3] D. Banerjee, P. A. Nair, R. Usbeck, and C. Biemann. Gett-qa: Graph embedding based t2t transformer for knowledge graph question answering. In C. Pesquita, E. Jimenez-Ruiz, J. McCusker, D. Faria, M. Dragoni, A. Dimou, R. Troncy, and S. Hertling, editors, *The Semantic Web*, pages 279–297, Cham, 2023. Springer Nature Switzerland.

[4] H. Bast and E. Haussmann. More accurate question answering on freebase. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, page 1431–1440, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2806416.2806472.

[5] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284:34–43, 2001. doi:10.1038/SCIENTIFICAMERICAN0501-34.

[6] S. Bonner, I. P. Barrett, C. Ye, R. Swiers, O. Engkvist, C. T. Hoyt, and W. L. Hamilton. Understanding the performance of knowledge graph embeddings in drug discovery. *Artificial Intelligence in the Life Sciences*, 2:100036, 2022. doi:10.1016/J.AILSCI.2022.100036.

[7] C. Chen, F. Tung, N. Vedula, and G. Mori. Constraint-aware deep neural network compression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 400–415, 2018.

[8] S. Chen, Q. Liu, Z. Yu, C.-Y. Lin, J.-G. Lou, and F. Jiang. ReTraCk: A flexible and efficient framework for knowledge base question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 325–336, Online, 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.acl-demo.39.

[9] Y. Chen, H. Li, G. Qi, T. Wu, and T. Wang. Outlining and filling: Hierarchical query graph generation for answering complex questions over knowledge graphs. *IEEE Transactions on Knowledge & Data Engineering*, 35(08):8343–8357, 2023. doi:10.1109/TKDE.2022.3207477.

[10] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, A. Castro-Ros, M. Pellat, K. Robinson, D. Valter, S. Narang, G. Mishra, A. Yu,

V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei. Scaling instruction-finetuned language models, 2022.

[11] R. Das, M. Zaheer, D. Thai, A. Godbole, E. Perez, J. Y. Lee, L. Tan, L. Polymenakos, and A. McCallum. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9594–9611, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.emnlp-main.755.

[12] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332, 2022.

[13] J. Devlin. Sharp models on dull hardware: Fast and accurate neural machine translation decoding on the CPU. In M. Palmer, R. Hwa, and S. Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2820–2825. Association for Computational Linguistics, 2017. doi:10.18653/v1/d17-1300.

[14] D. Diomedi and A. Hogan. Entity linking and filling for question answering over knowledge graphs. In *Natural Language Interfaces for the Web of Data (NLIWOD) Workshop*, 2022.

[15] F. Farazi, J. Akroyd, S. Mosbach, P. Buerger, D. Nurkowski, M. Salamanca, and M. Kraft. Ontokin: An ontology for chemical kinetic reaction mechanisms. *Journal of Chemical Information and Modeling*, 60:108–120, 2020. doi:10.1021/ACS.JCIM.9B00960.

[16] D. Fensel, U. Simsek, K. Angele, E. Huaman, E. Kärle, O. Panasiuk, I. Toma, J. Umbrich, A. Wahler, D. Fensel, and et al. Introduction: what is a knowledge graph? *Knowledge graphs: Methodology, tools and selected use cases*, pages 1–10, 2020.

[17] J. Gong, H. Shen, G. Zhang, X. Liu, S. Li, G. Jin, N. Maheshwari, E. Fomenko, and E. Segal. Highly efficient 8-bit low precision inference of convolutional neural networks with intelcaffe. In *Proceedings of the 1st on Reproducible Quality-Efficient Systems Tournament on Co-Designing Pareto-Efficient Deep Learning*, ReQuEST '18, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3229762.3229763.

[18] J. Gu, Z. Lu, H. Li, and V. O. Li. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany, 2016. Association for Computational Linguistics. doi:10.18653/v1/P16-1154.

[19] Y. Gu and Y. Su. ArcaneQA: Dynamic program induction and contextualized encoding for knowledge base question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1718–1731, Gyeongju, Republic of Korea, 2022. International Committee on Computational Linguistics.

[20] Y. Gu, S. Kase, M. Vanni, B. Sadler, P. Liang, X. Yan, and Y. Su. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488, 2021.

[21] Y. Gu, X. Deng, and Y. Su. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4928–4949, Toronto, Canada, 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.acl-long.270.

[22] R. Guha, M. T. Howard, G. R. Hutchison, P. Murray-Rust, H. Rzepa, C. Steinbeck, J. Wegner, and E. L. Willighagen. The blue obelisk - interoperability in chemical informatics. *Journal of Chemical Information and Modeling*, 46:991–998, 2006. doi:10.1021/ci050400b.

[23] R. Hirigoyen, A. Zouaq, and S. Reyd. A copy mechanism for handling knowledge base elements in SPARQL neural machine translation. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 226–236, Online only, 2022. Association for Computational Linguistics.

[24] K. M. Jablonka, L. Patiny, and B. Smit. Making the collective knowledge of chemistry open and machine actionable. *Nature Chemistry*, 14(4):365–376, 2022. doi:10.1038/s41557-022-00910-7.

[25] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713, 2018.

[26] Y. Jeon, C. Lee, E. Cho, and Y. Ro. Mr.biq: Post-training non-uniform quantization based on minimizing the reconstruction error. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12329–12338, 2022.

[27] M. Junczys-Dowmunt, K. Heafield, H. Hoang, R. Grundkiewicz, and A. Aue. Marian: Cost-effective high-quality neural machine translation in C++. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 129–135, Melbourne, Australia, 2018. Association for Computational Linguistics. doi:10.18653/v1/W18-2716.

[28] S. Kim, A. Gholami, Z. Yao, M. W. Mahoney, and K. Keutzer. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pages 5506–5518. PMLR, 2021.

[29] A. Kondinski, A. Menon, D. Nurkowski, F. Farazi, S. Mosbach, J. Akroyd, and M. Kraft. Automated rational design of metal-organic polyhedra. *Journal of the American Chemical Society*, 144:11713–11728, 2022. doi:10.1021/JACS.2C03402.

[30] N. Krdzavac, S. Mosbach, D. Nurkowski, P. Buerger, J. Akroyd, J. Martin, A. Menon, and M. Kraft. An ontology and semantic web service for quantum chemistry calculations. *Journal of chemical information and modeling*, 59:3154–3165, 2019. doi:10.1021/ACS.JCIM.9B00227.

[31] Z. Li and Q. Gu. I-vit: Integer-only quantization for efficient vision transformer inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17065–17075, 2023.

[32] P. Liang. Lambda dependency-based compositional semantics. *ArXiv*, 1309.4408, 2013.

[33] M. Q. Lim, X. Wang, O. Inderwildi, and M. Kraft. The world avatar—a world model for facilitating interoperability. In *Intelligent Decarbonisation: Can Artificial Intelligence and Cyber-Physical Systems Help Achieve Climate Mitigation Targets?*, pages 39–53. Springer, 2022.

[34] Y. Liu, S. Yavuz, R. Meng, D. Radev, C. Xiong, and Y. Zhou. Uni-parser: Unified semantic parser for question answering on knowledge base and database. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8858–8869, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-main.605.

[35] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[36] A. Menon, L. Pascazio, D. Nurkowski, F. Farazi, S. Mosbach, J. Akroyd, and M. Kraft. OntoPESScan: An ontology for potential energy surface scans. *ACS Omega*, 8(2):2462–2475, 2023. doi:10.1021/acsomega.2c06948.

[37] H. Mousavi, M. Loni, M. Alibeigi, and M. Daneshtalab. Dass: Differentiable architecture search for sparse neural networks. *ACM Trans. Embed. Comput. Syst.*, 22 (5s), 2023. doi:10.1145/3609385.

[38] National Center for Biotechnology Information. PubChem, 2023 (accessed October 15, 2023). URL `https://pubchem.ncbi.nlm.nih.gov/`.

[39] V. P. Nia, A. Ghaffari, M. Zolnouri, and Y. Savaria. Rethinking pareto frontier for performance evaluation of deep neural networks. *arXiv preprint arXiv:2202.09275*, 2022.

[40] Open AI. ChatGPT, 2023 (accessed October 15, 2023). URL `https://chat.openai.com/`.

[41] L. Pascazio, S. Rihm, A. Naseri, S. Mosbach, J. Akroyd, and M. Kraft. A chemical species ontology for data integration and knowledge discovery. *Journal of Chemical Information and Modeling*, 2023 (in press). doi:10.1021/acs.jcim.3c00820.

[42] J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of sparql. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. M. Aroyo, editors, *The Semantic Web - ISWC 2006*, pages 30–43, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[43] M. Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, 2018. Association for Computational Linguistics. doi:10.18653/v1/W18-6319.

[44] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[45] S. Ravishankar, D. Thai, I. Abdelaziz, N. Mihindukulasooriya, T. Naseem, P. Kapanipathi, G. Rossiello, and A. Fokoue. A two-stage approach towards generalization in knowledge base question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5571–5580, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.findings-emnlp.408.

[46] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing*, 2019.

[47] M. R. A. H. Rony, U. Kumar, R. Teucher, L. Kovriguina, and J. Lehmann. Sgpt: A generative approach for sparql query generation from natural language questions. *IEEE Access*, 10:70712–70723, 2022. doi:10.1109/ACCESS.2022.3188714.

[48] A. Saxena, A. Kochsiek, and R. Gemulla. Sequence-to-sequence knowledge graph completion and question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2814–2828, Dublin, Ireland, 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-long.201.

[49] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, 2017. Association for Computational Linguistics. doi:10.18653/v1/P17-1099.

[50] J. Senellart, D. Zhang, B. Wang, G. Klein, J.-P. Ramatchandirin, J. M. Crego, and A. M. Rush. Opennmt system description for wnmt 2018: 800 words/sec on a single-core cpu. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 122–128, 2018.

[51] Y. Shu, Z. Yu, Y. Li, B. Karlsson, T. Ma, Y. Qu, and C.-Y. Lin. TIARA: Multi-grained retrieval for robust question answering over large knowledge base. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Pro-*

cessing, pages 8108–8121, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-main.555.

[52] Y. Su, H. Sun, B. Sadler, M. Srivatsa, I. Gür, Z. Yan, and X. Yan. On generating characteristic-rich question sets for QA evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 562–572, Austin, Texas, 2016. Association for Computational Linguistics. doi:10.18653/v1/D16-1054.

[53] A. Talmor and J. Berant. The web as a knowledge-base for answering complex questions. In *North American Chapter of the Association for Computational Linguistics*, 2018.

[54] T. V. Veen. Wikidata. *Information technology and libraries*, 38:72–81, 2019.

[55] L. Wang, C. Yu, S. Salian, S. Kierat, S. Migacz, and A. F. Florea. Searching the deployable convolution neural networks for gpus. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12227–12236, 2022.

[56] Z. Yang, Y. Wang, X. Chen, B. Shi, C. Xu, C. Xu, Q. Tian, and C. Xu. Cars: Continuous evolution for efficient neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[57] X. Ye, S. Yavuz, K. Hashimoto, Y. Zhou, and C. Xiong. RNG-KBQA: Generation augmented iterative ranking for knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6032–6043, Dublin, Ireland, 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-long.417.

[58] X. Yin, D. Gromann, and S. Rudolph. Neural machine translating from natural language to sparql. *Future Generation Computer Systems*, 117:510–519, 2021. doi:10.1016/j.future.2020.12.013.

[59] D. Yu, S. Zhang, P. Ng, H. Zhu, A. H. Li, J. Wang, Y. Hu, W. Y. Wang, Z. Wang, and B. Xiang. Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[60] W. Zheng, J. X. Yu, L. Zou, and H. Cheng. Question answering over knowledge graphs: Question understanding via template decomposition. *Proc. VLDB Endow.*, 11(11):1373–1386, 2018. doi:10.14778/3236187.3236192.

[61] X. Zhou, D. Nurkowski, S. Mosbach, J. Akroyd, and M. Kraft. Question answering system for chemistry. *Journal of Chemical Information and Modeling*, 61:3868–3880, 2021. doi:10.1021/ACS.JCIM.1C00275.

[62] X. Zhou, D. Nurkowski, A. Menon, J. Akroyd, S. Mosbach, and M. Kraft. Question answering system for chemistry—a semantic agent extension. *Digital Chemical Engineering*, 3:100032, 2022. doi:10.1016/j.dche.2022.100032.

[63] X. Zhou, S. Zhang, M. Agarwal, J. Akroyd, S. Mosbach, and M. Kraft. Marie and BERT – a knowledge graph embedding based question answering system for chemistry. *ACS Omega*, 8:33039–33057, 2023.