A Question Answering System for Chemistry

Xiaochi Zhou,[†] Daniel Nurkowski,[‡] Sebastian Mosbach,[†] Jethro Akroyd,[†] and Markus Kraft[*,†,¶,§]

† *Department of Chemical Engineering and Biotechnology, University of Cambridge, Philippa Fawcett Drive, Cambridge, CB3 0AS, United Kingdom*

‡*CMCL Innovations, Sheraton House, Castle Park, Castle Street, Cambridge, CB3 0AX, United Kingdom*

¶ *School of Chemical and Biomedical Engineering, Nanyang Technological University, 62 Nanyang Drive, Singapore, 637459*

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

E-mail: mk306@cam.ac.uk

Abstract

This paper describes the implementation and evaluation of a proof-of-concept Question Answering system for accessing chemical data from knowledge graphs which offer data from chemical kinetics to chemical and physical properties of species. We trained question classification and named entity recognition models that specialize in interpreting chemistry questions. The system has a novel design which applies a topic model to identify the question-to-ontology affiliation to handle ontologies with different structures. The topic model also helps the system to provide answers with higher quality. Moreover, a new method that automatically generates training questions from ontologies is also implemented.

The question set generated for training contains 432989 questions under 11 types. Such a training set has been proven to be effective for training both the question classification model and the named entity recognition model. We evaluated the system using other KGQA systems as baselines. The system outperforms the chosen KGQA system answering chemistry-related questions. The QA system is also compared to the Google search engine and the WolframAlpha engine. It shows that the QA systems can answer certain types of questions better than the search engines.

Introduction

In recent years, many efforts have been made to create large knowledge graphs (KGs) to store data from various domains. A knowledge graph represents a collection of inter-connected descriptions of entities: real-world objects, events, or abstract concepts. The description of entities have formal semantics provided by ontologies.[1] As a result, machines can process information in KGs in an unambiguous manner. Also, previously isolated pieces of information are often inter-connected in a KG, allowing efficient retrieval of related data of entities.

Popular open-domain knowledge graphs include the Google KG,[2] the Wikidata KG,[3] and the DBpedia KG.[4] Those large-scale knowledge graphs contain significant amount of information related to the chemistry domain. For example, Wikidata and DBpedia contain basic physical and chemical properties of chemical species. There are also a number of ontologies specifically developed for representing chemical data. For example, OntoCAPE[5] (Ontology for Computer Aided Process Engineering) is developed modelling chemical processes. PubChemRDF[6] represents structures and metadata of chemical substances and compounds. OntoKin[7] is a chemical ontology specialized for representing chemical reaction mechanisms. OntoCompChem[8] is an ontology for quantum chemistry calculations. OntoSpecies[9] is an ontology designed to represent both generic and domain-specific information about species. OntoKin, OntoCompChem, and OntoSpecies contain both schema that defines concepts and relations and instantiated entities such as particular species and reactions.

The World Avatar (TWA) KG, which is also referred to as the J-Park Simulator (JPS) KG in some earlier literature,[10] is an attempt to create a general world model. TWA integrated and inter-connected OntoKin, OntoCompChem, and OntoSpecies ontologies. It is also connected to a part of the Wikidata KG, which provides basic physical and chemical properties such as heat capacity and mass of chemical substances.

With the chemical data stored in KGs, users can use semantic queries such as the SPARQL query to perform complex data retrieval accurately. For example, to find the chemical structure of all aromatic hydrocarbons with a molecular weight over a certain value. However, the barrier for constructing SPARQL queries is high as it requires expertise on KGs.

One solution to lower this barrier is to establish a Question Answering (QA) system.[11] A QA system allows end-users to pose natural language queries and accurately query information based on complex conditions.

A QA system utilizes machine learning models and natural language processing tools to interpret questions posed by end-users and convert the questions into machine-understandable queries. The common components of a QA system include classification of the questions, extraction of key components within the questions, mapping of the key components to their semantic representation, and query construction. The typical types of questions that can be answered include yes-or-no questions, factoid questions, list questions, and summary.[12]

A number of projects have been proposed to build both open-domain and domain specific QA systems. For example, Question Answering over Linked Data (QALD) [13] is a series of evaluation campaigns on question answering over linked data. Since 2011, 9 challenges have been held and yielded many projects for open-domain QA systems. Some recent projects include WDAqua,[14] ganswer2,[15] and QAKis.[16] A domain specific challenge for biomedical information QA systems is BioASQ.[12] Some prominent systems include HPI[17] and DeepMeSH.[18]

However, the open-domain QA systems built on top of open-domain KGs such as Wikidata, DBpedia, and YAGO[19] are not compatible with domain ontologies such as OntoCAPE, OntoKin, OntoCompChem, and OntoSpecies as they are of more complex structures compared with open-domain KGs. Also, to implement more accurate question classification and named entity recognition functions for a QA system to answer domain specific questions, it requires NLP models specifically trained for the domain of interest. As a result, the open-domain QA systems are not suitable for answering domain specific questions for chemistry. To the best of our knowledge, there is no QA system built for chemistry data.

In addition, some large-scale KGs such as TWA, consists of many domain ontologies of different specifications. However, it is common that there is overlapping of content between ontologies. In some cases, similar information can be offered by different ontologies but the information quality may vary due to their different specializations. For example, the Wikidata KG stores some information about reactions. However, OntoKin, as an ontology specifically made for chemical reactions, provides information of significantly higher quality than the Wikidata KG. Moreover, it is necessary to utilize different mechanism to construct the queries for ontologies with different structures in the same KG.

As a result, to construct the correct SPARQL queries and guarantee the answer quality, it is necessary to have a mechanism that chooses the preferred ontology and query construction mechanisms for different questions. There exist some noticeable works to identify the topics of questions. For example, Nie et al. proposed to use joint question-topic embedding learning to tag questions from community-driven question answering (cQA) websites. Similar works also include using sparse deep learning for disease inference from health-related questions.[21]

Therefore, this paper introduces our implementation and evaluation of a proof-of-concept QA system built on top of models specifically trained for chemistry-related questions. It also leverages topic modelling to identify the affiliation between questions and ontologies. A topic model is a statistical model for discovering the abstract "topics" that occur in a collection of documents. Such a model can also be used to analyse the composition of "topics" within a sentence such as a question.

However, the lack of training data is a major challenge for building such a QA system. The existing datasets such as LC-QuAD 2.0,[22] SimpleQuestionsWikidata,[23] and CSQA[24] mostly consist of general questions. As a result, they are not able to provide sufficient amount of chemistry-related training data for the NLP model specifically trained for a chemistry QA system. In addition, the manual creation of a sufficiently large training set is difficult and expensive, as it requires domain expertise. Therefore, we also propose a method to automatically generate and label training questions, leveraging the rich information contained in ontologies.

The contributions of this paper are: the introduction of a QA system for chemistry, the use of topic model to handle QA over multiple ontologies of different structures, and a fully automatic mechanism to generate training data for the NLP models using information from ontologies.

Background

In this section we review ongoing research efforts that we found important when implementing a KG-based natural language query interface, including the KGs TWA and Wikidata KG, as well as, natural language processing tools, and KG-based natural language query interfaces.

The World Avatar

Large cross-domain systems such as industrial symbioses, chemical plants, and cities are constituted by components such as power generators, storage tanks, and abstract industrial operations, which are from diverse domains. In order to achieve complex tasks including running simulations and optimizations, and coordination of multiple components, the relevant data, knowledge, and models must be integrated. However, the communication friction due to the heterogeneous conventions across domains hinders such an integration.

The World Avatar project aims to represent the physical world in all aspects. Such a concept is extended from the Digital Twin concept, which emphases on creating virtual representation of entities such as a device or an operation in industrial processes. The virtual representation will allow uniform integration between not only device and device but also devices and operations. Such uniform integration is similar to upgrading the Internet of Things to the Internet of Services and more.

One specific implementation of the World Avatar project is the J-Park Simulator (JPS) ,[25–27] which provides a data management common ground for those components and enable semantic interoperability so that cross-domain integration can be enabled. The JPS is now fully integrated in TWA. Figure 1 demonstrates the architecture of the JPS KG.

A number of ontologies from different domains, containing the definition and schema of concepts and relations as well as semantically described instances, form the terminology and instance layer of TWA. In addition, in order to update and maintain the large-scale KG over time, a number of agents, which carry out functions including data retrieval, simulation, and data update, are part of TWA and operate on top of it.

Knowledge graph

Ontologies are the backbones of knowledge graphs.[28] The T-boxes of the ontologies[29] define the schema of the KG. Meanwhile, the KG consists of the inter-connected A-boxes of the ontologies, which provide the semantically described instances of entities.

An ontology is a set of nodes including classes, properties, and instances denoted Internationalized Resource Identifiers (IRIs).[30] For example, in the Wikidata knowledge graph, the node describing the chemical species methane is defined by `https://www.Wikidata.org/wiki/Q37129`. IRIs provide both access to the declaration of the nodes as well as a universally unique identifier for the node. As a result, referring to nodes with IRIs solves the ambiguity problem when referring to entities or concepts with natural language terms. For instance, the word “methane” can also refer to a 1969 Italian movie but within the Wikidata ontology, the chemical species and the movie are assigned with a distinct IRI. Thus, due to the disambiguation, the data within a KGs are machine-readable and the conflicts between the convention of different data sources are eliminated. In addition, the nodes in KGs are connected and accessible through IRIs over the internet. All statements about the schema and the data are expressed in the form of semantic triples. A semantic triple is constituted by three entities, the subject, the predicate, and the object. For example, in the statement that declares the molecular weight of Benzene, the subject is "wd:Q2270", which is IRI of "Benzene". The predicate in the statement is "wdt:P2067", which is the IRI of "mass", while the object is the value 78.047. The object can also be a subject in another triple, connected to another entity such as the unit of the value. Therefore, a machine can navigate through the KGs and retrieve all the data related to a node with ease, even the nodes are stored distributed over the Internet, and such a feature enables complex queries over the KGs. Moreover, the ontologies not only share the common ground for data management, they are also interconnected following the Linked Data[31] principle, which means the entities in one ontology are connected to entities in other ontologies via IRIs if they are relevant. For example, a power plant node in the electricity system ontology can be connected to the node representing a city in where the plant is located within the Wikidata KG. Such as interconnection between ontologies allows mutual enrichment between them. To access the data in KGs, the main tool is SPARQL Protocol and RDF Query Language (SPARQL),[32] which allows deep search of data within the KG by providing conditions on relations or values. SPARQL queries can also be used to update, delete, or insert information in the KG and the structure of these queries is almost identical to the queries for searching for information.

With the features of KGs mentioned above, different chemistry-related ontologies including OntoKin representing chemical kinetic reaction mechanisms, OntoCompChem representing quantum chemistry calculations, and OntoSpecies representing chemical species[7] are seamlessly integrated into TWA.

Agents

Agents serve important roles in software environments nowadays. The main differences between an agent[33] and traditional software are, firstly, agents adopt a Service-Oriented Architecture.[34] From one aspect, agents are web services deployed on web servers that are accessible through the internet. As a result, the agents can be used in a distributed environment where digital resources are stored on different networked computers.[35] Secondly, different from web services, agents allow automatic management by computers including automatic discovery, composition, and execution.[36–38] The automatic management of agents is enabled by describing the details of the functions and protocols of agents with computer languages such as SOAP and WSDL. Many agents have been implemented to maintain and update KGs. To better manage and utilize the agents, they are described in the KG with agent ontologies. For example, in TWA, the agents are described by an agent ontology OntoAgent[39] so that the functionality and communication standards can be understood by machines.

Open-domain Knowledge Graphs

Wikidata is a collaboratively edited multilingual knowledge graph hosted by the Wikimedia Foundation, where the semantic web technologies[40] are leveraged to build a structured database. Wikidata not only offers a machine-readable database for knowledge in almost all domains, it also includes a very detailed ontological class hierarchy for chemical species. For example, the entity of "Benzene" is not only under the class "chemical compound" but also "class IB flammable liquid", "occupational carcinogen", "male reproductive toxicant", "developmental toxicant" and "carcinogen". Such a detailed classification for entities allows efficient selection of instances in their KG based on the classes, which is fairly useful for chemistry databases. In addition, the data comprehensiveness in Wikidata is also high for chemical species. Take the same example of Benzene, there are 212 properties connected to this node, including chemical formula, refractive index, molecular weight, ionization energy, autoignition temperature and so on. It also contains the identifiers of the species in other databases such as PubChem CID, ChemSpider ID, and CAS Registry Number, making the connection of the Wikidata chemical data to external databases.

The DBpedia[4,41] KG is another large-scale open-domain KG. DBpedia extracts structured content from the information from various Wikimedia projects. For example, the DBpedia project extracts data from infoboxes from Wikipedia pages. DBpedia also provides very detailed hierarchy information in its ontologies.

However, to the best of our knowledge, the Wikidata and the DBpedia KGs only offer basic physical and chemical properties of species, more detailed data such as thermochemistry data, phase change data, and IR spectrum are not included.

KG-based natural language query

To lower the barrier of accessing data in the KGs and other similar structured databases, many efforts have been made to develop natural language query interfaces. For example, Kaufmann et al.[42] proposed an early implementation of a natural language interface to query ontologies, which converts questions into SPARQL queries. In addition, it asks the user for clarification when there is ambiguity in the question. Wang et al.[43] proposed PANTO system that also translates questions into SPARQL queries by leveraging a set of off-the-shelf parsers. Tablan et al.[44] also introduced their QuestIO system which converts natural language questions into SeRQL language or other queries. Later, they presented another system FREyA. It includes an ontology-based lookup service to map natural language term into IRIs. Al-Zubaide and Issa[45] proposed an ontology-based ChatBot, where Artificial Intelligence Markup Language (AIML) is used to create a mapping of questions and queries in the form of categories and store the resulted scenarios. Numerous other research efforts in this field are worth mentioning. For example, Mishra and Khilwani[46] has proposed the QUASE ontology-Based Domain Specific Natural Language Question Answering System, which includes a question classification module to increase the accuracy of the query construction. Saha et al.[47] proposed ATHENA, which is an ontology-driven system for natural language querying over relational data stores.

The aforementioned implementations share similar designs. The query interfaces first parse the questions and separate and tag different components using Natural Language Processing (NLP) tools. For example, in the question “show me all the chemical species with molecular weight larger than 200”, an ideal parsing result will be “chemical species” as a “class”, “molecular weight” as an “attribute”, “larger” as a “comparison operator”, and “200” as a “numerical value”. The parsing result may also include the relation between the different components. Based on the parsing result, the next step is to convert the terms such as “molecular weight” into their Semantic representations, which are IRIs. The modules for mapping terms to IRIs are usually referred to as Ontology Lookup Service (OLS). One well-known implementation of the OLS system is proposed by Côté et al..[48] In addition, several upgrades[49,50] on the systems have been presented. The basic working mechanism is to create a mapping between the natural language labels of IRIs and the IRIs and return IRIs based on the string similarity between the IRI labels and the input terms. The last step of those systems is to fill the results obtained from the first two steps into SPARQL query templates.

Natural language processing toolkits

To support the interaction between computers and natural languages, many open-source natural language processing toolkits with different features have been developed. Existing toolkits include NLTK,[51] OpenNLP,[52] CoreNLP,[53] Gensim,[54] and Rasa framework.[55] To select the most suitable tools for each part of the query interface, these toolkits are evaluated in the context of chemistry and KGs. Some common functions offered by NLP toolkits are:

- **Part-of-speech (POS) tagging**: to mark the part of speech. For example, to tag “are” as “VBP (verb plural)”;

- **Text parsing**: to analyse a sentence and convert it into a tree structure indicating the syntactic relation between its components;
- **Named entity recognition (NER)**: to extract and classify objects appeared in a sentence. For example, in the sentence "what is Benzene", some NER models can extract "Benzene" and classify it as a "species".

Natural Language Toolkit (NLTK) is an open-source Python-based platform with common NLP functions including tokenization, stop words filtering, stemming, part-of-speech (POS) tagging, sentence parsing. One of the most prominent advantages is that it provides convenient interfaces to many corpora and lexicons. Therefore, the NLTK can be used without training any model, therefore it is suitable for the pre-processing of training documents or other simple tasks.

The Apache OpenNLP is a Java library that supports functions including tokenization, POS tagging, NER, parsing, and coreference resolution, which is to identify the noun phrases in the same sentences that refer to the same entity. For example, in the sentence "We like cats because they are cute", coreference resolution can identify that "cats" and "they" are referring to the same entities. Similar to NLTK, the OpenNLP offers common NLP functions but only model training.

Stanford CoreNLP is another Java-based NLP framework. Besides the aforementioned common NLP tasks, Stanford CoreNLP offers advanced sentiment analysis. CoreNLP is recognized as one of the most accurate NLP tools for NER and has been widely applied in NLP-related applications. In addition, CoreNLP allows users to expand the NER module by adding labelled entities to the module.

Gensim is the state-of-the-art Python-based open-source library for topic modelling, which discovers the abstract topic within a collection of documents. Gensim's topic modelling is built on top of Latent Dirichlet Allocation[56] (LDA), which is a form of unsupervised machine learning. Gensim's topic modelling library helps users to create topic models. The users only need to provide a set of training documents and set up several training parameters, the library will extract a number of abstract topic from the documents and train topic models that can identify the probability that a word belongs to one or more topics.

The Rasa framework is a conversational AI framework for building contextual assistants. Different from the aforementioned toolkits, the purpose of the Rasa framework is to provide all the tools necessary to build advanced Chatbots that naturally interact with human users. The Rasa framework contains four major modules: the interpreter module, the tracker module, the policy module, and the action module. The interpreter is built on top of NLP tools, to conduct tasks such as tokenization, text parsing, and NER. The result returned by the interpreter module contains the intent of the questions and the classified entities found in the sentence. The tracker module keeps track of the conversations and can provide information about the previous interactions. The policy module chooses the action from the Chatbot, such as ask the user for more information or query a database.

The Rasa interpreter module allows the users to train customized intent classification models and NER models. The users are required to provide questions where each question is labelled with their intent and each component in the sentence is highlighted and labelled with its classification. The intent classification models and NER models are built on top of the text embeddings technology.

Design and implementation

The QA system is implemented on top of TWA, which includes OntoKin, OntoCompChem, OntoSpecies, and the Wikidata KG. TWA is selected because it contains high quality chemistry data and the Wikidata KG is selected due to its very stable query service.

The overall working mechanism of our QA system is to convert a natural language question into a SPARQL query, which retrieves the desired information from a specific ontology in the KG. To achieve such purpose, there are several subsystems implemented:

- **Topic model agent**: to identify the affiliation between the question and the domain

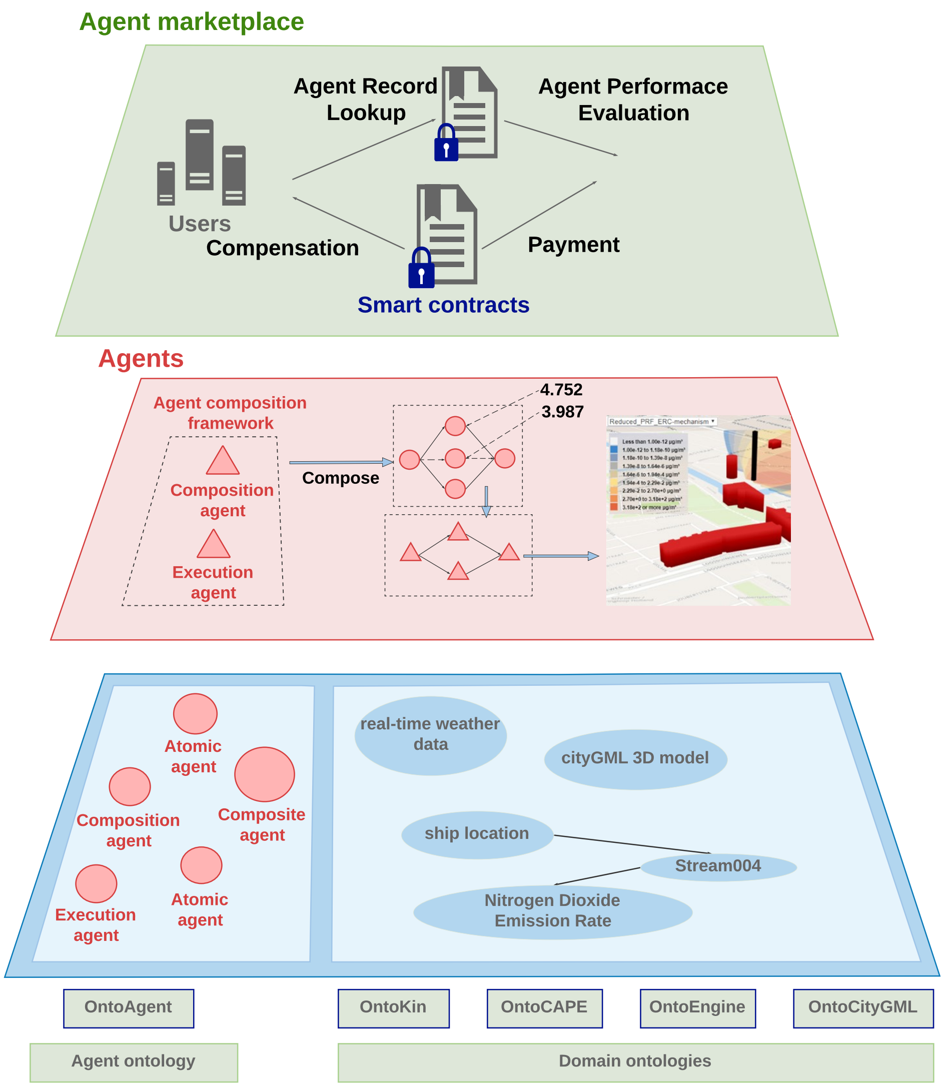
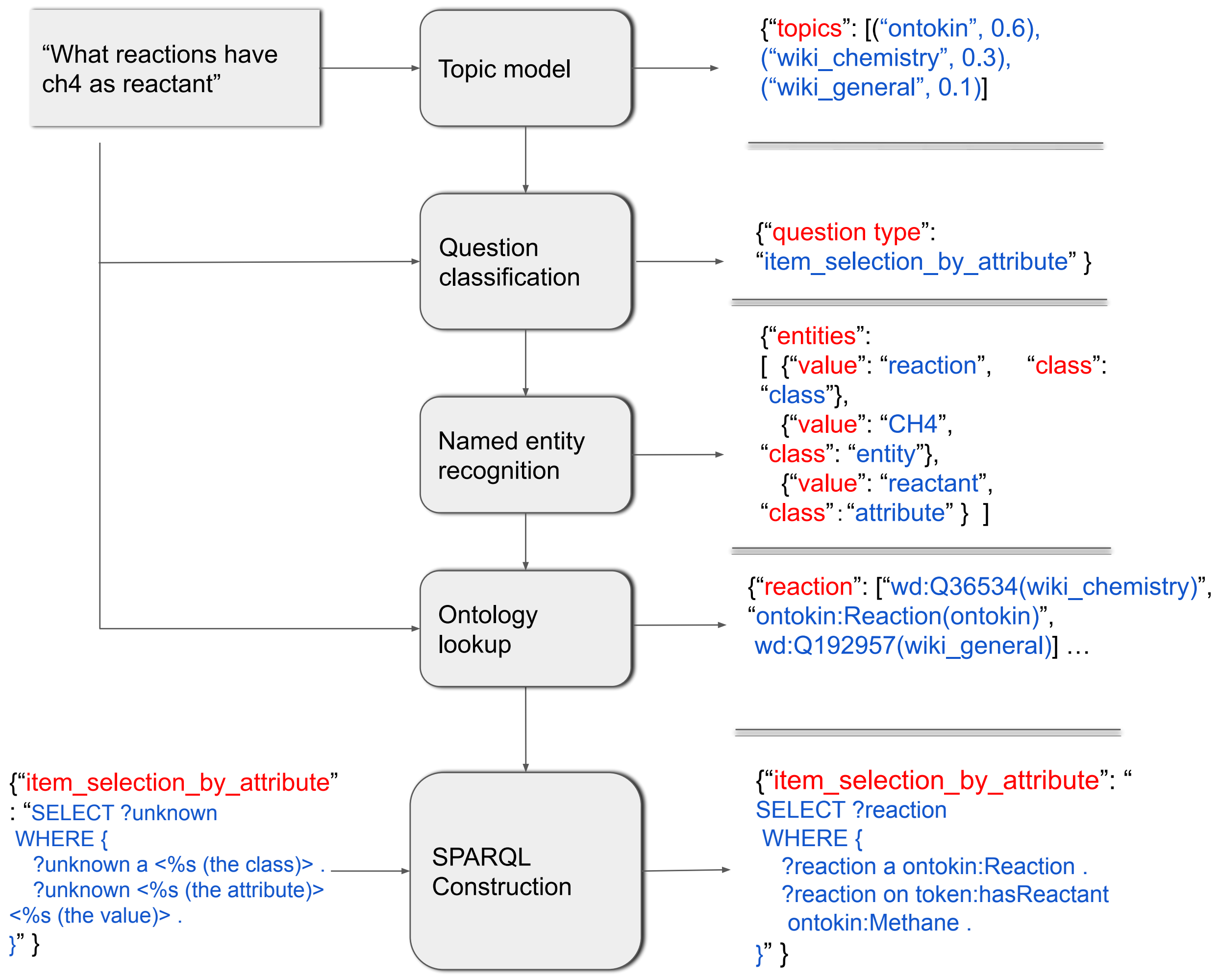


Figure 1: The architecture of the J-Park Simulator KG as part of TWA. The green boxes on the bottom are the terminology parts of the ontologies while the blue layer contains the instances of the ontologies, including the red circles which are the semantic descriptions of agents. The red layer contains the operating agents which are red triangles. On top of the agent layer, an agent marketplace is monitoring and evaluating the performance of the agents.

ontologies.

- **Question classification agent**: to identify the type of the question and map the question to the respective SPARQL query template;
- **Named entity recognition agent**: to extract the key components from the question;



"What reactions have ch4 as reactant"
Topic model
{"topics": [("ontokin", 0.6), ("wiki_chemistry", 0.3), ("wiki_general", 0.1)]
Question classification
{"question type": "item_selection_by_attribute" }
Named entity recognition
{"entities": [{"value": "reaction", "class": "class"}, {"value": "CH4", "class": "entity"}, {"value": "reactant", "class": "attribute" }]
Ontology lookup
{"reaction": ["wd:Q36534(wiki_chemistry)", "ontokin:Reaction(ontokin)", wd:Q192957(wiki_general)] …
SPARQL Construction
{"item_selection_by_attribute" : "SELECT ?unknown WHERE { ?unknown a <%s (the class)> . ?unknown <%s (the attribute)> <%s (the value)> . }" }
{"item_selection_by_attribute": " SELECT ?reaction WHERE { ?reaction a ontokin:Reaction . ?reaction on token:hasReactant ontokin:Methane . }" }

Figure 2: The general workflow of the query interfaces and the intermediate outputs of each module. The example of converting the question "what reactions have ch4 as reactant" into a SPARQL query is selected.

- **Ontology lookup agent**: to convert the key components in the question into IRIs;

Workflow

Figure 2 demonstrates the workflow of the QA system converting a natural language question into a valid SPARQL query.

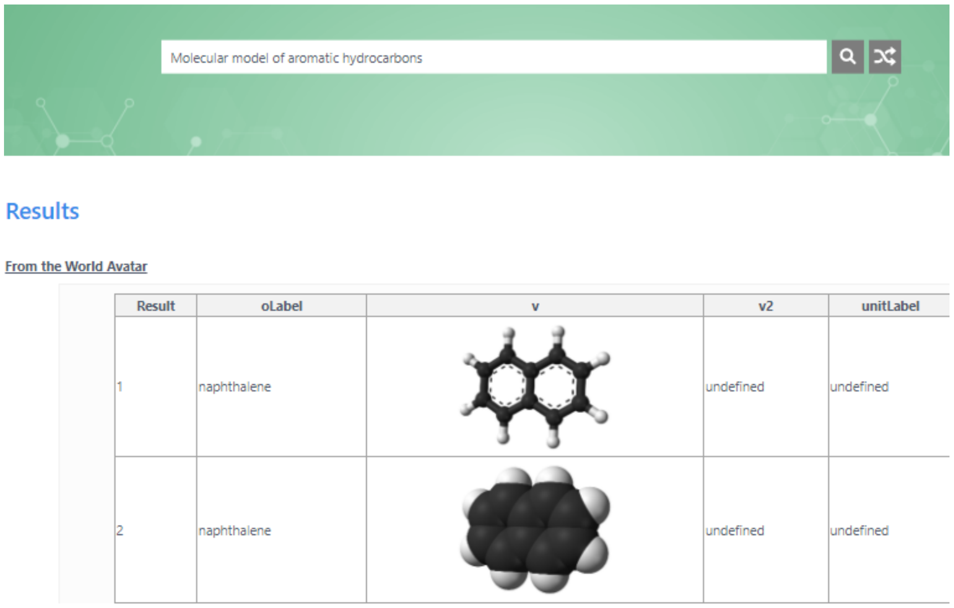
When the QA system receives a natural language question, the question classification agent will identify its question type and a SPARQL query template will be assigned for

```
SELECT ?oLabel ?v ?value ?unitLabel
WHERE {

 ?o wdt:P31  wd:<IRI for the class>  .
 ?o wdt:<IRI for the attribute>  ?v .
  OPTIONAL {
   ?o p:<IRI for the attribute>/psv:<IRI for the attribute>  ?v .
   ?v wikibase:quantityAmount ?value .
   ?v wikibase:quantityUnit ?unit .
  }
  SERVICE wikibase:label {bd:serviceParam wikibase:language "[AUTO_LANGUAGE], en" .}

} LIMIT 50
```

Figure 3: The SPARQL template for “batch_restriction_query_attribute” questions.



Molecular model of aromatic hydrocarbons

Results

From the World Avatar

Result	oLabel	v	v2	unitLabel
1	naphthalene		undefined	undefined
2	naphthalene		undefined	undefined

Figure 4: A screenshot of the QA website “Marie” for chemistry data. The website is powered by the QA system presented in this paper.

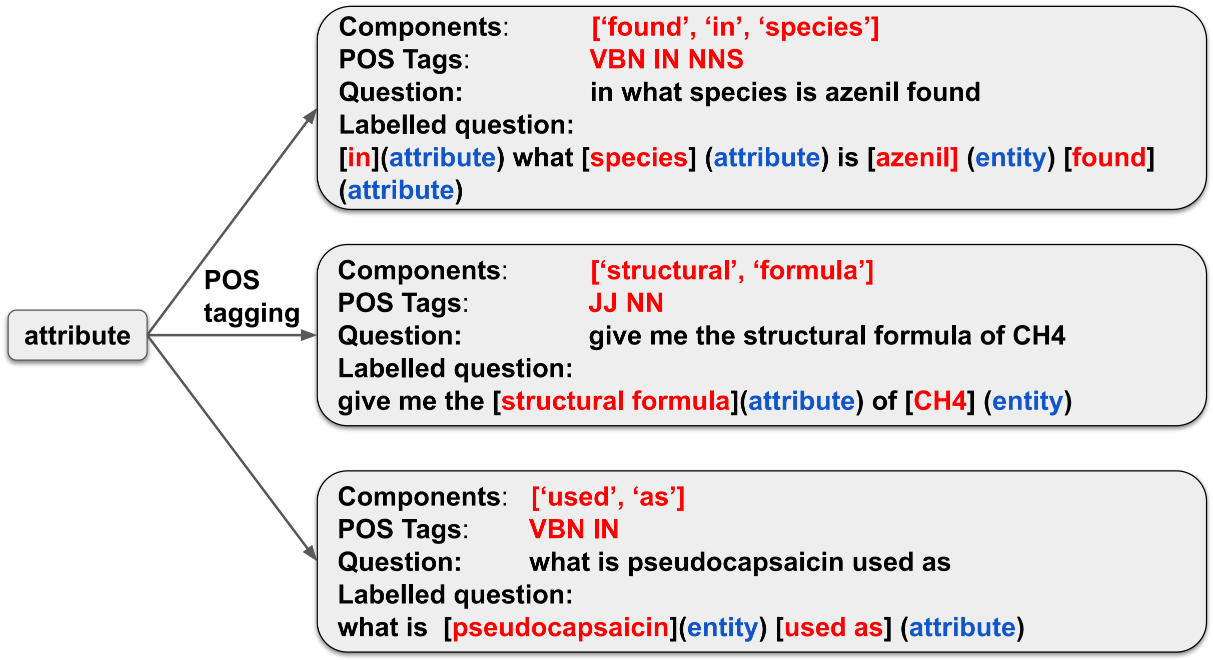

attribute
POS tagging
Components: ['found', 'in', 'species']
POS Tags: VBN IN NNS
Question: in what species is azenil found
Labelled question:
[in](attribute) what [species] (attribute) is [azenil] (entity) [found] (attribute)
Components: ['structural', 'formula']
POS Tags: JJ NN
Question: give me the structural formula of CH4
Labelled question:
give me the [structural formula](attribute) of [CH4] (entity)
Components: ['used', 'as']
POS Tags: VBN IN
Question: what is pseudocapsaicin used as
Labelled question:
what is [pseudocapsaicin](entity) [used as] (attribute)

Figure 5: The process of generating questions of different grammatical structures based on the Part-of-Speech tagging result of the attribute. Each black box represents a question example, where the first line is the texts of the attribute, the second line is the POS tags of the texts, and the last line is the questions generated with their components tagged.

each question type. If more than one possible question type are identified, a list of question types, which are ranked based on their confidence score, will be returned. Then the question classification agent passes the question and the question type to the named entity recognition agent. The named entity recognition agent will extract and classify the key components. Also, from the question classification result, the named entity recognition agent knows the numbers and the types of the key components within the question. Therefore, the named entity recognition agent can verify whether the recognition result matches the question type. If the match fails, the QA system will switch to the question type with the next highest confidence.

Then the named entity recognition agent will pass the key components to the ontology lookup agent, which searches the IRIs in the predefined dictionary based on the key components. As the ontology lookup agent conducts a fuzzy string match to find the IRIs, a ranked

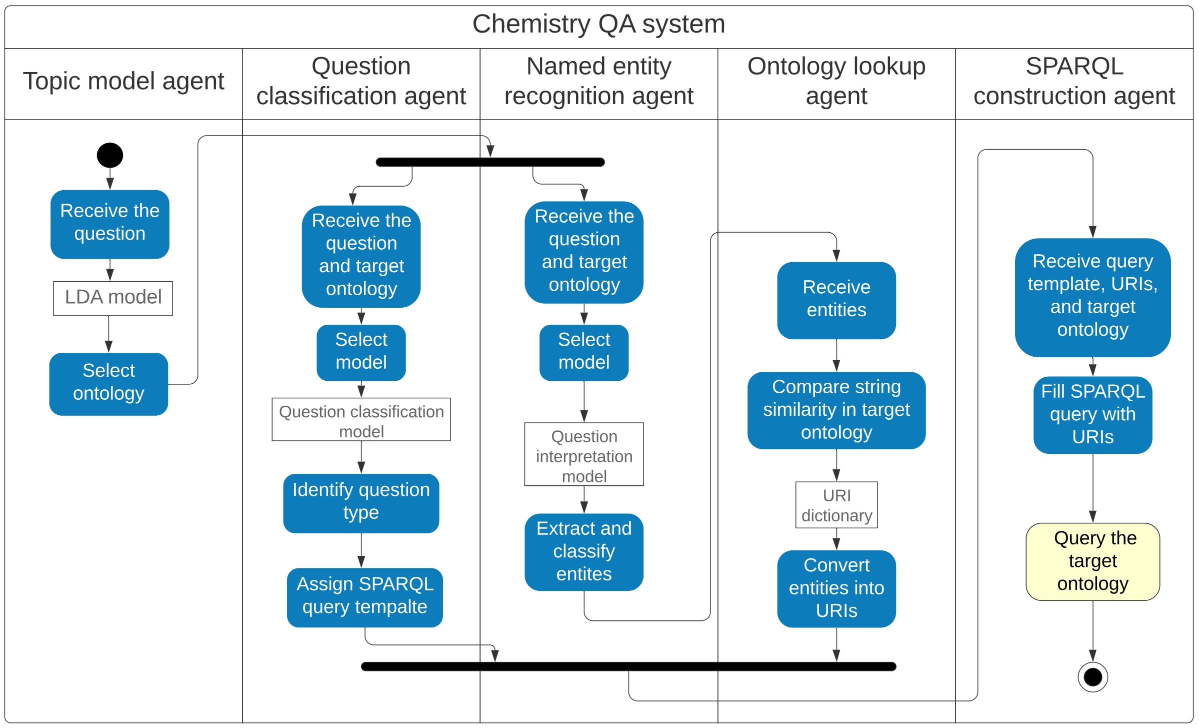

Chemistry QA system
Topic model agent
Question classification agent
Named entity recognition agent
Ontology lookup agent
SPARQL construction agent
Receive the question
LDA model
Select ontology
Receive the question and target ontology
Select model
Question classification model
Identify question type
Assign SPARQL query tempalte
Receive the question and target ontology
Select model
Question interpretation model
Extract and classify entites
Receive entities
Compare string similarity in target ontology
URI dictionary
Convert entities into URIs
Receive query template, URIs, and target ontology
Fill SPARQL query with URIs
Query the target ontology

Figure 6: A UML activity diagram that demonstrates the activity of the agents included in the QA system.

list of IRIs will be returned for each component. Moreover, as some entities or classes exist in multiple ontologies, multiple IRIs from different ontologies with identical scores might be found in the dictionary. As a result, the topic classification agent will help the ontology lookup agent to find the best IRI match. The ontology lookup agent will pass the components and the question to the topic classification agent to identify the affiliation between the question and the ontologies. Based on the affiliation scores, the ontology lookup agent will further rank the IRIs based on the string similarity and the topic affiliation.

At the last step, the SPARQL construction agent will construct the SPARQL query by filling the IRIs into the assigned SPARQL query template. Figure 3 provides an example of a SPARQL template. The constructed SPARQL query will be then sent to the respective SPARQL query.

A website named “Marie” is also implemented as the graphical user interface of the QA system. Figure 4 shows a screenshot of the website.

Table 1: Sample questions under the type "item_attribute_query"

Question list
1. show me the [production method](attribute) of [C11H14N2S]((entity))
2. what are the [frvd](attribute) of [naloxone](entity)
3. what are [described in url](attribute) of [mizolastine](entity)
4. the [English Vikidia ID](attribute) of [C5H8O2](entity)
5. show the [picture](attribute) of [sec-butyllithium](entity)
6. what is the [treats](attribute) of [C7H15NO3](entity)
7. what are the [award won](attribute) of Itraconazole(entity)
8. what [part](attribute) does [C4H6O](entity) include
9. list [Kemler code](attribute) of [C4H4O4](entity)
10. what is [CaN2O6](entity) a [subsystem of](attribute)
11. what is the [hardness](attribute) of [stearophanate](entity)
12. what are the [IOR](attribute) of [sterogyl](entity)
13. what is [L-ornithine](entity) the [form of](attribute)
14. the [manufacturer](attribute) of [immunocytophyte](entity)

Preliminary work

This section first describes the creation of training data and the training of three machine learning models on which the QA system is built: the question topic model, classification model, and the named entity recognition model. It then describes the creation of the term-IRI dictionary for the ontology lookup agent.

Data preparation

To train the question classification model, it requires a training set of questions labelled with their structural category. At the same time, the training of named entity recognition model requires a set of questions where their entities are underlined and labelled with their types. To train these two models, we created a set of training questions. Meanwhile, the entities in each question are underlined and labelled with their types. Table 1 shows some examples of the questions with highlighted entities.

Due to the high cost of manually creating the training set, we implemented a Python script to generate questions automatically. For each structural category, the script generates questions in a certain form. For example, the category "item_attribute_query", contains questions asking one attribute and one instance such as "what is CH4 used as". To create a question in this category, the script takes one instance and one of its relevant attribute in the ontologies and finds their natural language labels, for example, "used as" and "CH4". Then the script applies Part-of-Speech (POS) tagging to the attribute "used as". Since the attribute is made up by a verb in past particle tense (VBN) and a preposition (IN), it is put behind the instance "CH4" to form the grammatically correct question "what is CH4 used as". Figure 5 gives examples of how questions of different grammatical structures are created under the "item_attribute_query" category using POS tagging. In addition, for training the named entity recognition model, the script also label "used as" as "attribute" and "CH4" as "entity". In total, we generated 432989 questions under 11 categories and the questions contain 6 types of entities. The question categories and the entity types are listed in Appendix, are generated.

The training of the topic model requires a document for each ontology, which contains all the natural language words appearing in such ontology. We used SPARQL query to retrieve the labels and the comments of the nodes in the ontologies and put them in the training documents for topic model. In addition, some parts of the IRIs, such as OntoKin:ReactionMechanism, are also included in the documents after using regular expression to separate the words "reaction" and "mechanism". The documents are further processed with the Natural Language Toolkit (NLTK). The words are reduced into their stems and the stop words, which are the most frequently used English words, are removed.

Model training

The topic model is a Latent Dirichlet Allocation[56] (LDA), which is a form of unsupervised machine learning. The model is trained by the Gensim.models.ldamodel.LdaModel class provided by the Python-based Genism library. The documents are passed to the LdaModel function. There are four parameters for training the model: **"num_topic"**, **"passes"**, **"alpha"**, **"eta"**. **"num_topic"** is the number of topics expected. In our case, there are four ontologies: OntoCompChem (quantum calculation), OntoKin (reaction kinetic), OntoSpecies (chemical species), and Wikidata (chemical properties). The parameter **"passes"** defines the number of algorithm iterations. Several combinations of the hyper parameters "alpha" and "eta" are tested. The experiments show that a high alpha parameter (0.9) and a low eta parameter (0.1) are suitable for the four ontologies due to the prominent difference between the words that appear in the ontologies. Also, setting the passes to 1000 suits the relatively small sizes of the training documents. A higher passes will not significantly affect the accuracy of the system. However, we have not undertaken a systematic optimization at this stage for this proof-of-concept implementation.

The question classification model is a word embedding model.[57] The model embeds questions and the question types into the same space, where the questions are represented by a bag of words vector based on the term counts. The TensorFlow embedding module provided by the Rasa framework is used to train the question classification model. The configuration settings are by default provided by the Rasa framework. Based on the training set of questions with type labels, the question classification model is then trained.

The named entity recognition (NER) model is a conditional random field model[58] and the training is handled by the NER_CRF module provided by the Rasa framework. The model is chosen as it is suitable for training customized NER models. The NER_CRF module is implemented on top of conditional random fields (CRF) modeling method. In this module, CRFs are used for predicting the text sequences that use the contextual information to add information which will be used by the model to make a correct prediction.

Dictionary creation

The ontology lookup agent requires a dictionary which maps the terms to the IRIs. As a result, we utilized the natural language labels in the ontologies to build such a dictionary. We used SPARQL query to extract the rdfs:label and skos:altLabel value of all the nodes. However, due to the substantial size of the Wikidata, we used WDumper, which is a tool to create custom RDF dumps, to created a sub-graph of chemistry-related nodes of the Wikidata knowledge graph. The skos:altLabel value provides the synonyms for the nodes. For example, the skos:altLabel for methane contains words including "CH4", "marsh gas", "methyl hydride", "tetrahydridocarbon", and "tetrahydrogen monocarbide". We created separated dictionaries for classes, attributes, and entities. In total, 33,219 words are included in the dictionaries.

Agent description

This section will describe the implementation and working detail of the agents. The agents include the topic model agent, the question classification agent, named entity recognition agent, the ontology lookup agent, and the SPARQL construction agent. Figure 6 illustrates the interaction between the agents of the QA system.

Topic model agent

The topic model agent is implemented on top of the LDA topic model described in the model training section. The Gensim LdaModel module is used to carry out the function of identifying the question-ontology affiliation. When the agent receives the question, it first tokenizes the words of the question and use PorterStemmer to reduce the words to their stem forms. Then the agent uses the pre-loaded LDA model to identify the topic distribution of this word list. The topic distribution is represented by a list of topics, each assigned with a confidence score. The topic model agent will rank the topics according to their scores and pass the ranked topics to the agents in the next step. The topic model recognizes 4 topics, as a result, any topic that is outside the scope of the topic model will be given equal scores (0.25) for each of the four topics. Then the topic model agent will conclude that the question is out of the scope of the QA system since the question is not chemistry-related.

Question classification agent

The core of the question classification agent is the question classification model mentioned in the model training section. Rasa TensorFlow embedding module is used to load and run such a model. A set of SPARQL query templates are made in advance for each type of questions, where the specific IRIs are replaced by placeholders. The classification agent also loads the templates when initiated. The types and numbers of the components that each SPARQL query template expects are also encoded into the question classification agent. For example, "item_attribute_query" questions expect one attribute and one instance. By mapping the question received to the question space of the question classification model, the question classification agent produces one or more question types with confidence scores. Then the agent passes both the question types identified and expected types and numbers of the components to the next agent, which is the named entity recognition agent.

Named entity recognition agent

The named entity recognition is implemented on top of the NER model described in the model training section. The agent takes the question and the expected types and numbers of components provided by the Question classification agent as inputs. Based on the inputs, the named entity recognition agent extracts and classifies the key components, including attributes, instances, and classes, using the Rasa framework rasa_nlu module, which loads and runs the NER model. For chemical reactions, the agent also identifies any symbols, such as "==]" or "–>", that separate reactants and products and hence categorizes the species into reactants and products. Regular expression is applied for this function. The output of this agent is the key components and their type. For example, for question "What is the heat capacity of CH4", the output will be "heat capacity" as "attribute" and "CH4" as "instance". In addition, this agent verifies whether the types and numbers of the components in the results fulfills the expectation given by the question classification agent. If not, the agent will iterate to the next question type in the input.

Ontology lookup agent

The ontology lookup agent converts the key components passed by the named entity recognition agent into IRIs. When the ontology lookup agent receives a question component, it will iterate through all the terms in the dictionary mentioned in the model training section and calculate the string similarity between the terms and the question component. A Python library Fuzzywuzzy is used to calculate the string similarity based on their Levenshtein distance. According to the string similarity, the ontology lookup agent will return a ranked list of IRIs found in the dictionary. However, it is common that different ontologies contain the same attribute, instances, or classes. Therefore, this agent only select IRIs from the target ontology, which is already identified by the topic model agent. Then the key components extracted by the named entity recognition agent are converted into IRIs. The IRIs and their types, for example "OntoKin:Reaction" and "class" will be passed to the SPARQL construction agent.

SPARQL construction agent

The SPARQL construction agent receives the SPARQL query templates and IRIs with their types. Then the agent will create a list of possible combinations of SPARQL query templates and IRIs and rank the combinations based on the confidence scores of both the SPARQL query template given by the question classification agent and the scores of the IRIs given by the ontology lookup agent.

Due to the low speed of SPARQL queries, the SPARQL construction agent will send multiple SPARQL queries to the SPARQL endpoints at the same time via multi-threading. The valid result with the highest score will be returned.

Evaluation

For evaluation, we collected 100 evaluation questions from researchers in different fields in the Department of Chemical Engineering and Biotechnology in the University of Cambridge. In addition, 530 questions were manually created by other members in our research group knowing information stored in the knowledge graph. Questions that are out of the scope of this QA system, for example "CsI vs CsF stability", are excluded from the evaluation. In total, 361 questions are used in the evaluation. Ablation experiments are not conducted as the absence of any agent would disable the system entirely.

We present some typical questions that system can answer in Table 2 and some question-answer pairs are demonstrated in the Appendix.

We have evaluated this QA system together with two other QA systems that are available to us: QAnswer[59] and Platypus.[60] QAnswer and Platypus provide accessible online demos and operate on top of the Wikidata KG. Due to the lack of methods to automatically evaluate the results returned for these questions, the 361 question-answer pairs are examined manually and classified as correct or not correct. The evaluation result is presented in Table 3. The accuracy scores represent the percentage of questions that are correctly answered and "x" indicates that this type of question is not valid for the QA system.

The evaluation results show that this QA system outperforms the chosen QA systems in answering chemistry questions and can answer a wider range of questions.

Since other KGQA systems do not cover chemistry data such as thermodynamic and quantum calculation data, we also used the Google search engine and the WolframAlpha engine as baselines. Table 4 demonstrates the evaluation results. The evaluation result shows that this QA system can answer certain types of questions better than the search

Table 2: Typical questions that the QA system can answer.

Question list
Computational Quantum Chemistry: 1. Show me the vibration frequency of H2O2.
2. Find the gaussian files for C8H14.
3. What is the symmetry number of C8H14?
4. What is the spin multiplicity of C8H14?
5. Electronic energy of C2H2O2.
6. Show the formal charge of C3H6.
7. What is the geometry type of C2H2O2?
Kinetic and Thermodynamics 1. What is the lennard jones well depth of C2H2O2?
2. Give the polarizability of C2H2O2.
3. What is the dipole moment of C2H2O2?
4. Show the rotational relaxation collision number of C2H2O2.
Reactions and Mechanisms 1. What reaction produces H2 + OH?
2. Is the reaction H + H2O == H2 + OH reversible?
3. What reaction has CH4 as a reactant?
4. What mechanism contains CH4 + OH?
Class query 1. List the chemical formula of alkanol with heat capacity less than 15.
2. Show the mass of aromatic hydrocarbons with mass less than 170.
3. Aromatic hydrocarbons with mass less than 170.
4. Chemical formula of alkanol with heat capacity less than 15.
Query by SMILES 1. What is the molecular weight of C1CCCCC1?
2. Show me the molecular model of CH2=CHCHO.
3. Show me the ionization energy of C1=CC=CC=C1.
4. What is the heat capacity of C1=CC=CC=C1?

engines, as it uses the advantages of KGs.

In addition, by investigating the evaluation result in detail, we have noticed that the

Table 3: Comparison of performance between this QA system and other KGQA systems

	QAnswer	Platypus	Marie
simple questions over wikidata	0.35	0.23	**0.51**
class questions over wikidata	x	x	**0.50**
about questions over wikidata	x	x	**0.66**
rank questions over wikidata	x	x	**0.57**
class attribute questions over wikidata	x	x	**0.31**
numerical class questions over wikidata	x	x	**0.88**
thermodynamic questions over TWA	x	x	**0.45**
quantum questions over TWA	x	x	**0.46**
finding reactions by species over TWA	x	x	**0.62**
find mechanisms by reactions over TWA	x	x	**0.5**
reaction properties over TWA	x	x	**0.7**

Table 4: Comparison of performance between this QA system and search engine

	Google	WolframAlpha	Marie
simple questions over wikidata	**0.71**	0.57	0.51
class questions over wikidata	**0.75**	0.25	0.50
about questions over wikidata	**1.0**	**1.0**	0.66
rank questions over wikidata	0.14	0.28	**0.57**
class attribute questions over wikidata	0.16	**0.31**	**0.31**
numerical class questions over wikidata	x	0.38	**0.88**
thermodynamic questions over TWA	0.06	x	**0.45**
quantum questions over TWA	0.23	0.13	**0.46**
finding reactions by species over TWA	0.23	0.46	**0.62**
find mechanisms by reactions over TWA	x	x	**0.5**
reaction properties over TWA	0.1	x	**0.7**

Table 5: Performance of agents

	Topic model	Question classification	Named Entity Recognition	Ontology Lookup
F1	0.82	0.93	0.925	0.90
Recall	x	x	0.947	x
Precision	x	x	0.904	x

queries over Wikidata ontologies have better performance than the queries over the JPS ontologies. By analysing the errors, we conclude that the better performance of Wikidata ontologies is due to their shallow structure, comparing to the deeper structure of JPS ontologies. A shallow ontology has a simpler structure. The Wikidata and DBpedia ontologies are examples of shallow ontologies, where the data are directly connected to the instances by one relation. By comparing the evaluation results between the JPS ontologies and Wikidata ontologies, it is easier to generate the SPARQL queries for the shallower ontologies

Moreover, the topic model agent, question classification agent, named entity recognition agent, and ontology lookup agent are evaluated separately. Recall and precision scores are provided when applicable. The result can be found in Table 5.

Error analysis

In table 5, we have provided the evaluation results on the individual agents. In this section, we will provide some detailed analysis on the errors in the agents with some examples.

Question classification agent

More than 60 percent of wrong question classification results comes from the mismatching between the “batch_attribute_query” and “item_attribute_query”, where the former contains one attribute and one class and the later contains one attribute and one species. For example, the question “what is the melting point of naphthalene” is classified as “batch_attribute_query” question.

For a text embedding model, it is very difficult to distinguish these two types of questions. However, in most of the cases, the named entity recognition model has successfully identified and classified the classes and species. As a result, the named entity recognition result is used to rectify wrong results from the question classification agent.

Named entity recognition agent

The errors from the named entity recognition agent can be categorized into two types: failed entity extraction and wrong entity classification.

An example of failed entity extraction is that in the question “what is the odor threshold of concentrated acetic acid”, the NER agent only extracted “odor” instead of “odor threshold” as the attribute.

An example of wrong entity classification is that in the question “show the structure diagrams of ionic diatomics”, the class “ionic diatomics” is identified as a species.

Nearly all the cases of failed entity extraction and wrong entity classification contains entities not included in the ontologies and hence the training questions. As a result, we suppose the solution to improve the NER agent is to extend the vocabulary in the ontologies.

Ontology lookup agent

The first type of failed cases in ontology lookup agent is that the ontologies does not include such entity in the first place. The examples include attributes such as “h-bond donors” and “graph properties”.

The second type of failed cases is that the agent failed to collect the vocabularies from the ontologies. For example, the attribute “gene atlas image” exist in the Wikidata KG as “Gene Atlas Image (P692)”. The attribute is not included in the dictionary because such an attribute is usually attached to large proteins and the large proteins are included in the KG dumps we created.

Conclusion

This paper presents the the novel design and the evaluation of a proof-of-concept QA system for chemical data over KGs. This QA system integrates a topic model, a named entity recognition, and a question classification model customized for the chemistry domain.

The evaluation results show that this QA system, with the customized models for chemistry, has better performance than some other KGQA systems in answering chemistry questions and can handle a wider range of question types. The QA system also answered certain types of questions better than the Google and WolframAlpha engines, as this QA system uses the advantage of KGs. The results also show that the training questions automatically generated from the ontologies can be used to train NLP models with decent performance.

A topic model is integrated into this QA system and has successfully handled multiple ontologies with different ontologies and provide answers of higher qualities.

Although the accuracy of the models can be further improved, this proof-of-concept QA system has shown its capability to answer several types of chemistry-related questions of relatively high complexity.

Future work

According to the error analysis on the individual agents, the agents can be further improved. For the topic model agent, although a set of hyper parameters have been tested for training the model, systematic experiments were not conducted to optimize the model. As a result, we will further optimize this model through more thorough experiments on the hyper parameters. Both the ontology lookup agent and the named entity recognition agent can be improved by extending the coverage of vocabularies from the ontologies. For the question classification agent, we would like to experiment on methods other than bag of words, such as the word2vec model,[61] to improve the performance of the agent.

In the evaluation set of questions, there is a substantial number of questions that are out of the scope of the QA system, we also aim to answer those questions by adding training data and SPARQL templates to the system in the future.

Data and Software Availability

The training data, evaluation data, evaluation results are available in the GitHub repository https://github.com/cambridge-cares/TheWorldAvatar under subdirecotry JPS_Chatbot.

Training data

The training data for the question classification model and the named entity recognition model are available in the nlu.md files under "UI/data" and "UI/source/rasa_jps/data" directory.

The training data for the topic model is available in the "LDA/corpus" directory and can be reproduced using the "make_lda_corpus.py".

Evaluation data

The evaluation questions and their query results are provided by the Excel file "Evaluation_results.xls".

Software

All the third-party software used in this system are free and available.

The Python environment suitable for operating the QA system is Python3.7 and all the Python libraries required and their versions are listed in the file "JPS_Chatbot/requirements.txt". All the packages from NLTK 3.5 also need to be downloaded and installed.

Geckodriver 0.24.0 and Firefox 73.0 are required to enable access to the WolframAlpha engine and the Google search engine.

A docker solution is also available for quick deployment of the system under the directory "JPS_Chatbot/docker". The latest version of Docker Desktop is required.

Question categories and component types

Question category	**Purpose**
select_mechanism_by_reaction	find mechanism by specifying reaction
select_reaction_by_species	find reactions by reactants and/or products
query_reaction_property	find properties such as reaction rate of a reaction
query_quantum_chemistry	query computational quantum properties
query_thermodynamic	query kinetic and thermodynamic properties
batch_restriction_query	query an attribute of all instances of of one class
item_attribute_query	query an attribute of a specific instance
batch_restriction_query_numerical	find instances which meet a numerical restriction
rank_query_attribute	sort instances of a certain class by the value of an attribute
about_query_attribute	query the description of an instance
batch_query	list all instances of a certain class
Component type	**Example**
attribute	electric dipole moment
class	aromatic hydrocarbons
entity	CH4
comparison	lower than
numerical_value	200

Examples of question-answer pairs

Question	**Answer**
what is the vapor pressure of H2O2	5 millimeter of mercury
what is the heat capacity of benzene	16.157 joule per mole kelvin
what is the molecular weight of c1ccccc1	84.094 dalton

References

(1) Gruber, T. *Encyclopedia of Database Systems*; Springer US: Boston, MA, 2009; pp 1963–1965.

(2) The Google Knowledge Graph. *Strategic direction (Bradford, England)* **2014**, *30*, 15–17.

(3) Van Veen, T. Wikidata. *Inform Technol Libr* **2019**, *38*, 72–81.

(4) Lehmann, J.; Isele, R.; Jakob, M.; Jentzsch, A.; Kontokostas, D.; Mendes, P. N.; Hellmann, S.; Morsey, M.; van Kleef, P.; Auer, S.; Bizer, C. DBpedia – A Large-scale, Multilingual Knowledge Base Extracted From Wikipedia. *Semant. Web* **2015**, *6*, 167–195.

(5) Morbach, J.; Yang, A.; Marquardt, W. OntoCAPE—A Large-scale Ontology For Chemical Process Engineering. *Eng. Appl. Artif. Intell.* **2007**, *20*, 147–161.

(6) Fu, G.; Batchelor, C.; Dumontier, M.; Hastings, J.; Willighagen, E.; Bolton, E. PubChemRDF: Towards the Semantic Annotation of PubChem Compound and Substance Databases. *J. Cheminformatics* **2015**, *7*, 1–15.

(7) Farazi, F.; Akroyd, J.; Mosbach, S.; Buerger, P.; Nurkowski, D.; Salamanca, M.; Kraft, M. OntoKin: An Ontology for Chemical Kinetic Reaction Mechanisms. *J. Chem. Inf. Model.* **2020**, *60*, 108–120.

(8) Krdzavac, N.; Mosbach, S.; Nurkowski, D.; Buerger, P.; Akroyd, J.; Martin, J.; Menon, A.; Kraft, M. An Ontology and Semantic Web Service for Quantum Chemistry Calculations. *J. Chem. Inf. Model.* **2019**, *59*, 3154–3165.

(9) Farazi, F.; Krdzavac, N. B.; Akroyd, J.; Mosbach, S.; Nurkowski, D.; Menon, A.; Kraft, M. Linking Reaction Mechanisms and Quantum Chemistry: An Ontological Approach. *Comput. Chem. Eng.* **2020**, *137*, 106813.

(10) Menon, A.; Krdzavac, N. B.; Kraft, M. From Database to Knowledge Graph — Using Data in Chemistry. *Curr. Opin. Chem. Eng.* **2019**, *26*, 33 – 37.

(11) Unger, C.; Freitas, A.; Cimiano, P. An Introduction to Question Answering over Linked Data. Reasoning Web International Summer School. 2014; pp 100–140.

(12) Tsatsaronis, G.; Balikas, G.; Malakasiotis, P.; Partalas, I.; Zschunke, M.; Alvers, M. R.; Weissenborn, D.; Krithara, A.; Petridis, S.; Polychronopoulos, D.; Almirantis, Y.; Pavlopoulos, J.; Baskiotis, N.; Gallinari, P. An Overview of the BioASQ Large-scale Biomedical Semantic Indexing and Question Answering Competition. *BMC Bioinf.* **2015**, *16*, 138.

(13) Ngomo, N. 9th Challenge on Question Answering over Linked Data (QALD-9). *Language* **2018**, *7*.

(14) Diefenbach, D.; Singh, K.; Maret, P. WDAqua-core0: A Question Answering Component for the Research Community. Semantic Web Challenges. 2017; pp 84–89.

(15) Zou, L.; Huang, R.; Wang, H.; Yu, J. X.; He, W.; Zhao, D. Natural Language Question Answering over RDF: A Graph Data Driven Approach. Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. New York, NY, USA, 2014; p 313–324.

(16) Cabrio, E.; Cojan, J.; Gandon, F.; Hallili, A. Querying Multilingual DBpedia with QAKiS. The Semantic Web: ESWC 2013 Satellite Events. Berlin, Heidelberg, 2013; pp 194–198.

(17) Schulze, F.; Schüler, R.; Draeger, T.; Dummer, D.; Ernst, A.; Flemming, P.; Perscheid, C.; Neves, M. HPI Question Answering System in BioASQ 2016. Proceedings of the Fourth BioASQ workshop. 2016; pp 38–44.

(18) Peng, S.; You, R.; Wang, H.; Zhai, C.; Mamitsuka, H.; Zhu, S. DeepMeSH: Deep Semantic Representation for Improving Large-scale Mesh Indexing. *Bioinformatics* **2016**, *32*, i70–i79.

(19) Suchanek, F.; Kasneci, G.; Weikum, G. Yago: a core of semantic knowledge. Proceedings of the 16th international conference on world wide web. 2007; pp 697–706.

(20) Nie, L.; Li, Y.; Feng, F.; Song, X.; Wang, M.; Wang, Y. Large-Scale Question Tagging via Joint Question-Topic Embedding Learning. *ACM Trans. Inf. Syst.* **2020**, *38*, 1–23.

(21) Nie, L.; Wang, M.; Zhang, L.; Yan, S.; Zhang, B.; Chua, T.-S. Disease Inference from Health-Related Questions via Sparse Deep Learning. *IEEE Trans. Knowl. Data. Eng.* **2015**, *27*, 2107–2119.

(22) Dubey, M.; Banerjee, D.; Abdelkawi, A.; Lehmann, J. *The Semantic Web – ISWC 2019*; Lecture Notes in Computer Science; Springer International Publishing, 2019; pp 69–78.

(23) Diefenbach, D.; Tanon, T.; Singh, K.; Maret, P. Question answering benchmarks for wikidata. ISWC 2017. 2017.

(24) Saha, A.; Pahuja, V.; Khapra, M. M.; Sankaranarayanan, K.; Chandar, S. Complex Sequential Question Answering: Towards Learning to Converse Over Linked Question Answer Pairs with a Knowledge Graph. **2018**,

(25) Zhou, L.; Zhang, C.; Karimi, I. A.; Kraft, M. An Ontology Framework Towards Decentralized Information Management for Eco-industrial Parks. *Comput. Chem. Eng.* **2018**, *118*, 49 – 63.

(26) Eibeck, A.; Lim, M. Q.; Kraft, M. J-Park Simulator: An Ontology-based Platform for Cross-domain Scenarios in Process Industry. *Comput. Chem. Eng.* **2019**, *131*.

(27) Farazi, F.; Salamanca, M.; Mosbach, S.; Akroyd, J.; Eibeck, A.; Aditya, L. K.; Chadzynski, A.; Pan, K.; Zhou, X.; Zhang, S.; Lim, M. Q.; Kraft, M. Knowledge Graph Approach to Combustion Chemistry and Interoperability. *ACS Omega* **2020**, *5*, 18342–18348.

(28) Chen, H.; Ji, H.; Sun, L.; Wang, H.; Qian, T.; Ruan, T. *Knowledge Graph and Semantic Computing : Semantic, Knowledge, and Linked Big Data*; 2016.

(29) Eschenbach, C. *Formal Ontology in Information Systems*; Frontiers in artificial intelligence and applications Formal ontology in information systems; IOS Press: Amsterdam, 2008.

(30) https://www.w3.org/International/O-URL-and-ident.html.

(31) Bizer, C.; Heath, T.; Berners-Lee, T. *Semantic Services, Interoperability and Web Applications: Emerging Concepts*; IGI Global, 2011; pp 205–227.

(32) World Wide Web Consortium's RDF Data Access Working Group, SPARQL Query Language for RDF. 2008; `https://www.w3.org/TR/rdf-sparql-query/`Last accessed June 24, 2021.

(33) Griffiths, N.; Chao, K.-M. *Agent-based service-oriented computing*; Springer, 2010.

(34) Barry, D. K. *Web services, service-oriented architectures, and cloud computing the savvy manager's guide*, 2nd ed.; The Savvy Manager's Guides; Morgan Kaufman, 2013.

(35) Shamma, J. *Cooperative control of distributed multi-agent systems*; John Wiley & Sons, 2008.

(36) Li, B.; Tang, X.; Lv, J. The Research and Implematation of Services Discovery Agent in Web Services Composition Framework. 2005 International Conference on Machine Learning and Cybernetics. 2005; pp 78–84.

(37) Greenwood, D.; Buhler, P.; Reitbauer, A. Web Service Discovery and Composition using the Web Service Integration Gateway. 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service. 2005; pp 789–790.

(38) Sycara, K.; Paolucci, M.; Ankolekar, A.; Srinivasan, N. Automated Discovery, Interaction and Composition of Semantic Web Services. *J. Web Semant.* **2003**, *1*, 27 – 46.

(39) Zhou, X.; Eibeck, A.; Lim, M. Q.; Krdzavac, N. B.; Kraft, M. An Agent Composition Framework for the J-Park Simulator - a Knowledge Graph for the Process Industry. *Comput. Chem. Eng.* **2019**, *130*, 106577.

(40) Berners-Lee, T.; Hendler, J.; Lassila, O. Semantic Web. *Sci. Am.* **2001**, *284*, 34–43.

(41) Lehmann, J.; Isele, R.; Jakob, M.; Jentzsch, A.; Kontokostas, D.; Mendes, P.; Hellmann, S.; Morsey, M.; Van Kleef, P.; Auer, S.; Bizer, C. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semant. Web* **2014**, *6*.

(42) Kaufmann, E.; Bernstein, A.; Zumstein, R. Querix: A natural language interface to query ontologies based on clarification dialogs. 5th International Semantic Web Conference (ISWC 2006). 2006; pp 980–981.

(43) Wang, C.; Xiong, M.; Zhou, Q.; Yu, Y. PANTO: A Portable Natural Language Interface to Ontologies. The Semantic Web: Research and Applications. Berlin, Heidelberg, 2007; pp 473–487.

(44) Tablan, V.; Damljanovic, D.; Bontcheva, K. A Natural Language Query Interface to Structured Information. The Semantic Web: Research and Applications. Berlin, Heidelberg, 2008; pp 361–375.

(45) Al-Zubaide, H.; Issa, A. A. OntBot: Ontology based chatbot. International Symposium on Innovations in Information and Communications Technology. 2011; pp 7–12.

(46) Mishra, V.; Khilwani, N. QUASE: AN Ontology-Based Domain Specific Natural Language Question Answering System. *Int J Recent Technol Eng.* **2019**,

(47) Saha, D.; Floratou, A.; Sankaranarayanan, K.; Minhas, U. F.; Mittal, A. R.; Özcan, F. ATHENA: an ontology-driven system for natural language querying over relational data stores. *Proceedings of the VLDB Endowment* **2016**, *9*, 1209–1220.

(48) Côté, R. G.; Jones, P.; Apweiler, R.; Hermjakob, H. The Ontology Lookup Service, a Lightweight Cross-platform Tool for Controlled Vocabulary Queries. *BMC Bioinformatics* **2006**, *7*, 97–97.

(49) Côté, R. G.; Jones, P.; Martens, L.; Apweiler, R.; Hermjakob, H. Ontology Lookup Service: More Data and Better Tools for Controlled Vocabulary Queries. *Nucleic Acids Res.* **2008**, *36*, W372–W376.

(50) Côté, R.; Reisinger, F.; Martens, L.; Barsnes, H.; Vizcaino, J. A.; Hermjakob, H. The Ontology Lookup Service: Bigger and Better. *Nucleic Acids Res.* **2010**, *38*, W155–W160.

(51) Loper, E.; Bird, S. NLTK: The Natural Language Toolkit. Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1. USA, 2002; p 63–70.

(52) Apache Software Foundation, OpenNLP Natural Language Processing Library. 2014; `http://opennlp.apache.org/` Last accessed June 24, 2021.

(53) Manning, C.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S.; McClosky, D. The Stanford CoreNLP Natural Language Processing Toolkit. Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations. 2014; pp 55–60.

(54) Řehůřek, R.; Sojka, P. Software Framework for Topic Modelling with Large Corpora. Proceedings of the LREC 2010 Workshop on New Challenges for NLP. 2010; pp 45–50.

(55) Rasa Technologies, Open source conversational AI — Rasa - Rasa. 2020; `https://rasa.com/` Last accessed June 24, 2021.

(56) Blei, D. M.; Ng, A. Y.; Jordan, M. I. Latent dirichlet allocation. Advances in neural information processing systems. 2002; pp 601–608.

(57) Chiu, B.; Crichton, G.; Korhonen, A.; Pyysalo, S. How to Train good Word Embeddings for Biomedical NLP. Proceedings of the 15th Workshop on Biomedical Natural Language Processing. Berlin, Germany, 2016; pp 166–174.

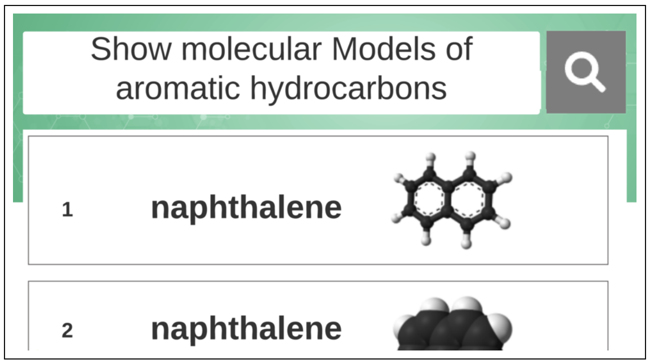
(58) Sutton, C.; McCallum, A. An Introduction to Conditional Random Fields. *Mach. Learn.* **2011**, *4*, 267–373.

(59) Diefenbach, D.; Migliatti, P. H.; Qawasmeh, O.; Lully, V.; Singh, K.; Maret, P. QAnswer: A Question Answering Prototype Bridging the Gap between a Considerable Part of the LOD Cloud and End-Users. The World Wide Web Conference. New York, NY, USA, 2019; p 3507–3510.

(60) Pellissier Tanon, T.; de Assunção, M. D.; Caron, E.; Suchanek, F. M. Demoing Platypus – A Multilingual Question Answering Platform for Wikidata. The Semantic Web: ESWC 2018 Satellite Events. Cham, 2018; pp 111–116.

(61) Church, K. W. Word2Vec. *Nat. Lang. Eng* **2017**, *23*, 155–162.

Graphical TOC Entry


Show molecular Models of aromatic hydrocarbons
1 naphthalene
2 naphthalene