Knowledge and Information Systems (2023) 65:5503–5522
https://doi.org/10.1007/s10115-023-01919-1

REGULAR PAPER

Ontology extension with NLP-based concept extraction for domain experts in catalytic sciences

Alexander S. Behr[1] · Marc Völkenrath[1] · Norbert Kockmann[1]

Received: 9 January 2023 / Revised: 23 May 2023 / Accepted: 21 June 2023 /
Published online: 15 July 2023

Abstract
Ontologies store semantic knowledge in a machine-readable way and represent domain knowledge in controlled vocabulary. In this work, a workflow is set up to derive classes from a text dataset using natural language processing (NLP) methods. Furthermore, ontologies and thesauri are browsed for those classes and corresponding existing textual definitions are extracted. A base ontology is selected to be extended with knowledge from catalysis science, while word similarity is used to introduce new classes to the ontology based on the class candidates. Relations are introduced to automatically reference them to already existing classes in the selected ontology. The workflow is conducted for a text dataset related to catalysis research on methanation of CO_2 and seven semantic artifacts assisting ontology extension by domain experts. Undefined concepts and unstructured relations can be more easily introduced automatically into existing ontologies. Domain experts can then revise the resulting extended ontology by choosing the best fitting definition of a class and specifying suggested relations between concepts of catalyst research. A structured extension of ontologies supported by NLP methods is made possible to facilitate a Findable, Accessible, Interoperable, Reusable (FAIR) data management workflow.

Keywords Ontology · Natural language processing · Automated ontology annotation · Information extraction · CO_2 methanation · Catalytic conversion

1 Introduction

In the current research data management, interconnection of the data produced and its interpretation are essential for comprehensible deductions of new knowledge. Research data need

✉ Alexander S. Behr
alexander.behr@tu-dortmund.de

Marc Völkenrath
marc.voelkenrath@tu-dortmund.de

Norbert Kockmann
norbert.kockmann@tu-dortmund.de

[1] Laboratory of Equipment Design, Faculty of Biochemical and Chemical Engineering, TU-Dortmund University, Emil-Figge-Straße 68, 44139 Dortmund, North-Rhine-Westphalia, Germany

to be FAIR (Findable, Accessible, Interoperable, and Reusable) by humans and machines in order to make proper use of data recorded in experiments, e.g., in electronic laboratory notebooks [1, 2]. While a researcher can easily grasp and interpret semantics expressed in texts using their implicit knowledge [3], a machine cannot perform this without having a representation of such knowledge embedded. Here, ontologies are used to describe implicit knowledge in an explicit way as they represent explicit specifications of conceptualizations [4]. Ontologies are informatic constructs used to represent relations among classes, such as *catalyst* or *reactor*.

As classification is an important concept of ontologies, the hierarchic sorting of the classes in turn represents the backbone of the ontologies. While the connection of classes within ontologies is important for their definition, short definition sentences (definition strings) are used as class annotation. This helps humans using the ontology to define and understand the classes of the ontology properly. Not only ontologies can be used to obtain definition strings for classes. Thesauri also provides classes with respective definition strings, such as the NCIT [5]. While they do not necessarily have semantic relations between their concepts like ontologies, they often contain more concepts and respective definition strings than ontologies.

For a domain expert who wants to represent the domain knowledge in an ontology, the hurdle to include ontology classes in the correct form into an ontology might be quite challenging and time consuming. Being experts in certain scientific fields, domain experts might also omit some knowledge because it is considered as trivial. Extending an ontology for own needs often is tedious work [6, 7]; thus, approaches are desired to simplify extension of ontologies and reduce consumed time for domain experts in order to raise acceptance of ontologies.

Since already existing ontologies do not necessarily contain all classes essential to describe the respective knowledge domain, an automated extension of ontologies is desirable. In addition, plenty of information is presented in scientific research in textual form, e.g., research papers by many domain experts. Those research papers contain a high number of domain-specific vocabulary. Using techniques from Natural Language Processing (NLP), in turn, can help to automate the setup of ontologies based on unstructured (natural) text as contained in research papers [8]. Exemplarily, by using Part of Speech (POS) tagging, nouns can be sorted out automatically from a given text and afterward be brought to their nominative singular form by lemmatizing.

While methods exist to extract ontologies from documents fully automatically, they usually provide ontologies that are not really useful for further reuse [9]. The ConTrOn (continuously trained ontology) project shows how user feedback can be integrated by a human-in-the-loop system [10, 11]. Here, a domain-specific ontology is augmented automatically and extended on basis of textual data and external sources of knowledge such as Wikidata and WordNet [12]. While the approach represents a solution to integrate information from data sheets to ontologies, the extraction of knowledge and relations between ontology classes from text is missing. In addition, a comparison of classes and their definitions with WikiData is done, while a comparison of classes and their definitions with other ontologies also would make sense. This is due to the fact that other ontologies also might contain knowledge not represented in WikiData, as ontologies focus more on expert knowledge.

The scope of this work is to use NLP techniques to extract vocabulary relevant to a domain of knowledge represented in a set of scientific papers. This vocabulary then is annotated by definitions derived from existing semantic artifacts (such as ontologies and thesauri) to help domain experts in later steps with sorting out the classes best fitting to the domain of knowledge. In addition, NLP is used to assist domain experts by including suggested classes automatically into an existing ontology and suggesting semantic relations between the classes based on text vectorization models of the texts. As classes should be only defined once to avoid ambiguities, already existing definitions of the added classes are included in the resulting extended ontology to later aid domain experts with selection of the most fitting definition to the automatically added classes. Thus, words necessary to describe a knowledge domain are included in a holistic, automated way into an ontology by including knowledge from a variety of scientific papers on a certain topic of interest.

2 Methodological background

This section describes the text dataset and the semantic artifacts used later to apply the workflow. Furthermore, the vectorization with Word2Vec is explained as its cosine similarity and *min_count* parameter serve as key classificators of later results.

2.1 Text dataset

The dataset deals with scientific publications focusing on catalytic methanation reactions. Here, a total of 25 research papers and three review papers are collected on research topics of methanation of CO_2. Besides continuous text, the dataset also contains other data, such as figures, diagrams, tables, and chemical formulas. In addition, the header and footer of pages often contains text with no further domain-specific information. Thus, preprocessing of the scientific publications focuses on extraction of token of the continuous text of the text dataset and omitting data waste. The method of preprocessing is described further in Sect. 3.1. The publications used as text dataset in this work are presented in Table A1 in Appendix A.

2.2 Semantic artifacts

For extension and annotation of ontologies, five ontologies and two thesauri are selected based on the set of ontologies deemed as important to the catalysis research domain by the NFDI4Cat project [1, 13, 14]. The Allotrope Foundation Ontology (AFO) [15], Chemical Entities of Biological Interest (CHEBI) [16], and Chemical Methods Ontology (CHMO) [17] are closely related to the chemical domain and contain concepts related to chemical experiments in laboratories. In contrast, the BioAssay Ontology (BAO) [18] focuses on biological screening assays and their results. While the scope of the BAO might not be intuitively fitting to the chosen text dataset, certain concepts are contained in the BAO such as chemical roles of substances (e.g., catalyst), which also play a role in the text dataset. Similar to that, the scope of the Systems Biology Ontology (SBO) [19] is system biology and computational modelling. Similar to the BAO, it is chosen as it also contains relations regarding substances and also general laboratory contexts, which also are contained in the text dataset.

In addition to these ontologies, two thesauri are used: the IUPAC Compendium of Chemical Terminology (IUPAC-Goldbook) [20] and the National Cancer Institute Thesaurus (NCIT) [5]. They cover vast amounts of chemical species and domain-specific words of the chemical domain of knowledge while also providing definition strings for the respective words. In order to be processed properly, all ontologies and the NCIT were used in the OWL file format in RDF/XML syntax and converted to OWL (RDF/XML), when only available in, e.g., TTL-serialization using Protégé [21]. IUPAC-Goldbook was used in json-file format

Table 1 Semantic artifacts used in this work

Ontology	Classes
AFO	2894
BAO	7514
CHEBI	176,873
CHMO	3084
SBO	694
Thesaurus	Concepts
IUPAC-Goldbook	7038
NCIT	166,212

as provided by the homepage [20]. The semantic artifacts discussed and used in this work are listed in Table 1 along with the number of classes or concepts they contain.

2.3 Vectorization with Word2Vec

After preprocessing the data, it is further used to get semantic similarity of the token extracted. For this, the algorithm Word2Vec implemented in the python module *gensim* is used [22]. It vectorizes words to learn relations between token and thus, represents a statistical method. Using the preprocessed text as input, Word2Vec creates a vocabulary, vectorizing each word to a vector of user defined length. While a longer vector corresponds to a higher dimension of the vector space used for the vectorization, it also results in longer computational time resulting in a trade-off between computational time and expressivity of the vectors [23]. The similarity of two concepts can be calculated with the help of the cosine similarity by calculating the cosine of the angle φ between two vectors $\vec{a}$ and $\vec{b}$ using the equation

$$\cos(\varphi) = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|} \tag{1}$$

resulting in a value close to one for token close to each other and close to minus one for token far away from each other. Because this is a statistical method, the frequency of occurrence of the token within the text corpus is important to consider. This is reflected in the Word2Vec parameter *min_count* setting the number of occurrences in the text corpus, a token must have at least to be considered by the model. The higher this number is set, the smaller the overall considered number of words gets; thus, the model focuses only on the most occurring words. A lower *min_count* is more prone to include token based on, e.g., typing errors or are those of less relevance to the overall domain of knowledge represented in the text corpus.

3 Method

To obtain information from scientific papers, the text corpus first needs to be extracted and preprocessed to be viable in further steps. Part of Speech tagging (POS-tagging) is used to extract only nouns as candidates for new ontological classes. Searching for these extracted concepts (token) in already existing semantic artifacts (ontologies or thesauri) yields token annotated with definition strings and linkage to the respective semantic artifact, the definition was taken from. To extend an already existing ontology with concepts based on the found

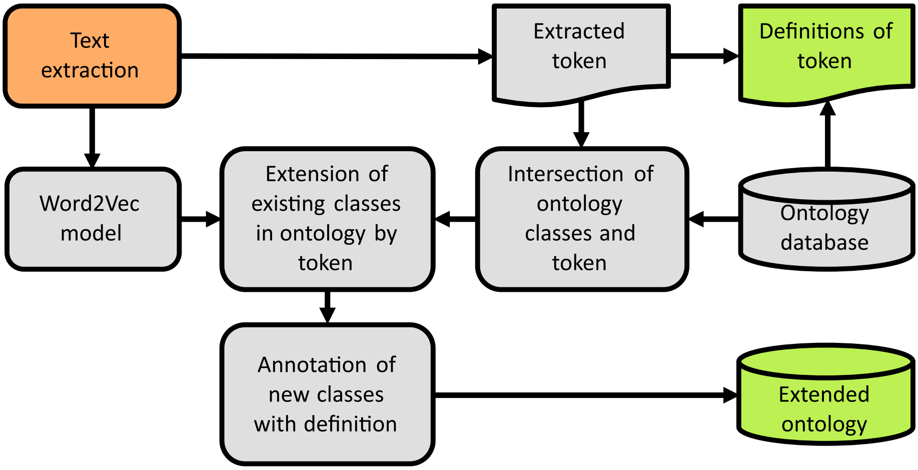

Text extraction
Extracted token
Definitions of token
Word2Vec model
Extension of existing classes in ontology by token
Intersection of ontology classes and token
Ontology database
Annotation of new classes with definition
Extended ontology

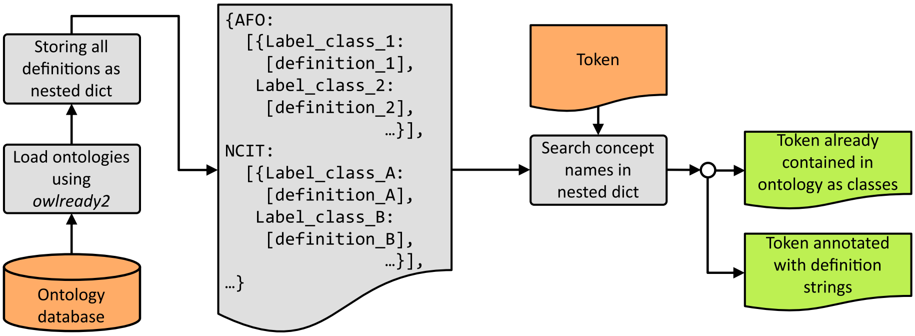
Fig. 1 Overall workflow conducted in this work to extract token from text, supply them with definitions based on ontologies and extend ontologies with new classes. The red box denotes the start of the workflow, while the output boxes are colored green (Color figure online)

token, a Word2Vec model is trained that vectorizes the text data. This in turn allows to output tokens with small cosine similarity to the already contained classes of an ontology and introducing those as new classes in the ontology. In addition, relations to denote semantic relation of these classes are posed, to connect the already contained ontology class to the automatically created classes based on Word2Vec. This overall workflow is depicted in Fig. 1 with the start of the workflow denoted in red and the output of the workflow in green. The following sections explain the main three steps of this general workflow in more detail. First, the text extraction is explained in detail, as the text corpus first needs to be extracted and preprocessed to be useful in further steps. Then, POS-tagging and search of the token in already existing ontologies takes place to annotate the extracted token. In the final step, the extension of an ontology by new classes based on the text dataset is explained.

3.1 Text extraction

Data from the text dataset contains, besides textual information, also information that is either non-textual or meaningless. Non-textual information, such as figures, can be neglected to reduce the file size. Text fragments without further domain-specific information also can be deleted to get a more condensed text dataset.

Thus, all figures, tables, and diagrams that do not contain complete sentences are removed first by hand with acrobat reader [24] and using the python module pdfminer [25]. Annotations and tables containing text in bullet point form are considered individually. Furthermore, lists such as references, table of figures, and table of nomenclature are removed, as these usually represent a list of individual words and symbols that do not reflect any context or relations. However, definition directories containing technical terms explained by short sentences are not removed, since they can contain relevant information. Subsequently, textual content that occurs repeatedly is removed, such as a DOI contained in the footer of each page or the journal name in the header of each page. These have no informative value and would negatively influence the creation of the model. Captions are also removed, since their


Storing all definitions as nested dict
Load ontologies using *owlready2*
Ontology database
Token
Search concept names in nested dict
Token already contained in ontology as classes
Token annotated with definition strings

```
{AFO:
  [{Label_class_1:
    [definition_1],
   Label_class_2:
    [definition_2],
              …}],
NCIT:
  [{Label_class_A:
    [definition_A],
   Label_class_B:
    [definition_B],
              …}],
…}
```

Fig. 2 Workflow of the code constructed for the annotation of extracted token. The red elements denote the input of the workflow, while the output boxes are colored green (Color figure online)

information content is marginal and repeat often without enhancement of the textual dataset (such as“Introduction” or“Conclusion”). Those cleaned files of the dataset are read in as strings using python code as a singular string such that each dataset contains a single string. The module SpaCy [26] is used to apply POS-tagging. This transforms the read-in string into a nested list, where each sentence is represented as list entry in a separate list. Using interpunction and space characters as separators, token are extracted and lemmatized using the vocabulary *en_core_web_sm*. This categorizes each word contained in each sentence regarding its lexical category (e.g., noun, verb, number,...).

3.2 Annotation of extracted token

As ontology classes are mostly nouns, only token with categories “noun” and “proper noun” are retained from the dataset and used in further procedures. Thus, a search of those token in ontologies is performed to determine the amount of token contained in each ontology as a class. The result helps to decide, which ontology can be taken as basis in further extension steps. Further help is provided by extraction of definitions of classes contained as string values in the ontologies, enabling for an easy determination of the best definition by domain experts in later steps.

To choose a fitting ontology to the dataset and enrich it by the concepts gathered by preprocessing, existing definitions of token contained in the ontologies should be known. Thus, python code is produced, which loads ontologies based on a local database using owlready2 [27]. Then, all class labels as well as their definition strings are read in from the ontologies and stored as key-value pairs in dictionaries. Nested dictionaries are used to store all classes and their definitions of a single ontology in a dictionary with the ontology name as key and the dictionary containing class names and their definitions as value. Token found by text extraction, as discussed in Sect. 3.1, is read in, and the dictionary is browsed for those token in class names. Finally, the number of found token per ontology can be accessed. In addition, the token is stored in a table along with the respective definitions, each assigned to its source ontology for later review of domain experts. The workflow of the code constructed for the annotation of extracted token is depicted in Fig. 2. The red elements denote the needed input of the workflow, i.e., the ontology database and the token obtained by text extraction, while the output boxes are colored green.

3.3 Extension of an ontology by new classes based on text dataset

The Word2Vec model is trained on the textual data obtained by the methods discussed in Sect. 3.1. Following [23], a vector size of 300 was set. While the Word2Vec model could be used for hierarchic clustering, the resulting clusters would not yield hierarchies in an ontological, semantic way. This is due to the nature of relations between token extracted by vectorization of concepts. As the text-clusters contain semantic similarities of words important for domains of knowledge, no classification and hierarchical information is obtained from the Word2Vec model. Thus, hierarchical clustering with, e.g., dendrograms, would not necessarily yield classifications (ontology classes and respective subclasses) of concepts. However, Word2Vec is able to give token with high cosine similarity to an initial input concept.

To use this functionality of similar token, the output of the workflow presented in Sect. 3.2 is used. The workflow not only annotates token of a text dataset with definitions contained in ontologies, but also can be used to output which token already are contained in each investigated ontology.

Picking the ontology with most common classes, these already contained classes are used as input for the Word2Vec model trained on the text dataset. The model then is used to retrieve the closest n token regarding cosine similarity of the input word. This is accompanied by a threshold value, restricting the amount of output token also with regards to the minimal cosine similarity allowed. This would allow for, e.g., setting a necessary minimal cosine similarity of 0.999, which would in turn only yield token very close to the input, while a minimal similarity of 0.8 would also include broader token, farther away in the vector space. As those token are most similar to the already contained ontology class, the ontology class and the token retrieved in this way by Word2Vec are assumed to have some kind of a semantic relationship.

If a token output by Word2Vec in this way is not already contained in the ontology, a new class has to be created, reflecting the token. To have an overarching class of newly included classes, not yet defined properly by semantic means, a class called *w2vConcept* is created as a subclass of *owl:Thing* class. Token output by the Word2Vec model and not yet contained in the ontology are then created as class. In addition, they are set to be subclasses of the also automatically created class *w2vConcept*, which in turn is set as subclass of the ontology root class *owl:Thing*. This is done to help in the later revision of the automatically created classes as they are more easy to find using an ontology editor, e.g., Protégé, when listed as subclass of the same class. Furthermore, this ensures that the integration of new classes does not disturb the semantic integrity of the ontology. The unique classes are also connected via an automatically created relationship to the classes deemed as similar by the Word2Vec model. This object property is called *conceptually related to* and is intended to ease the later definition of the exact relation between the two classes. To annotate the classes with missing definition strings, the workflow presented in Sect. 3.2 is used to search for definition strings of the newly created classes in other semantic artifacts. The code cannot decide by itself which definition might be more fitting when multiple definition strings are found. Thus, each definition string obtained is listed in a separate *rdfs:comment* of the class along with a note on the source of the definition.

After storing the resulting extended ontology, domain experts thus can go through newly added classes and easily accept or neglect the classes and modify the *conceptually related to* relation to a relation more fitting. This workflow of code to extend an ontology automatically is depicted in Fig. 3. The ontology used as input is denoted red, while the extended ontology, which poses the output of the workflow, is colored green.

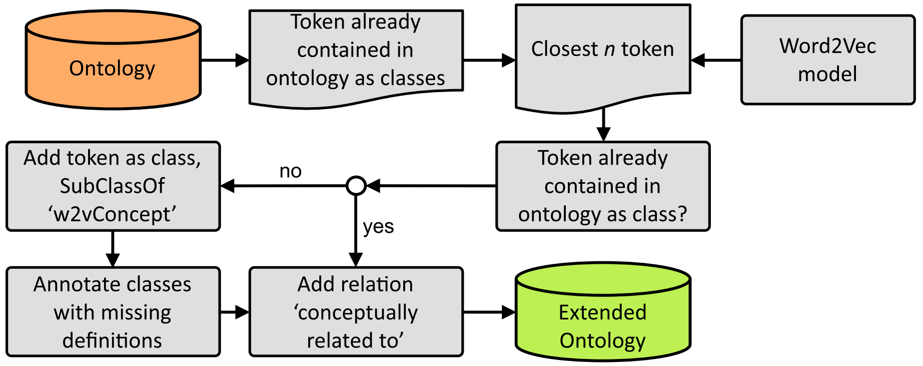

Ontology
Token already contained in ontology as classes
Closest *n* token
Word2Vec model
Token already contained in ontology as class?
no
yes
Add token as class, SubClassOf 'w2vConcept'
Annotate classes with missing definitions
Add relation 'conceptually related to'
Extended Ontology

Fig. 3 Workflow of code to extend an ontology by new classes based on text dataset. The ontology used as input is denoted red, while the extended ontology, which poses the output of the workflow, is colored green (Color figure online)

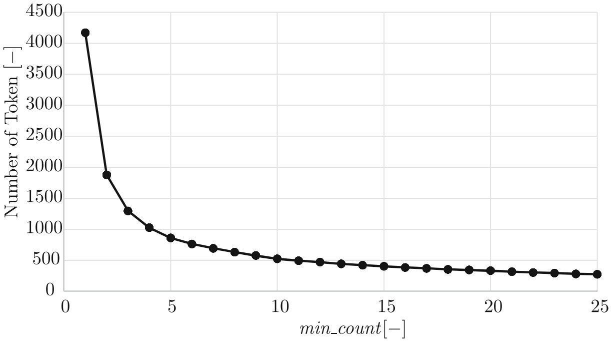


Fig. 4 Number of token obtained from the text dataset of 28 scientific papers for different *min_count* parameters

4 Results and discussion

The textual data of 28 scientific texts are preprocessed and extracted according to Sect. 3.1. This yields a dataset of overall 858,014 symbols which result in 4,170 noun token identified for further use in the workflows proposed in Sect. 3. Applying different *min_count* parameters in the range *min_count* = [1...25] yields different amounts of token as shown in Fig. 4. While higher *min_count* parameters yield lower amounts of token, the token contained is deemed the more important ones, as they occur more often in the dataset.

The resulting sets of token are then used as concept names to search for fitting classes in the seven semantic artifacts proposed in Sect. 2.2. This yields the number of token already contained in the respective ontology as classes as well as textual definitions of the classes in an automated way. In addition to this, the count of classes already contained can be used to suggest the ontology most fitting with regards to the respective text dataset.

Table 2 lists the resulting numbers of found classes in semantic artifacts of the performed annotation for six different *min_count* in the range [1...100]. Each token only needs to be annotated with a textual definition at least once; thus, the overall sum of annotated token is calculated for each set of token. Thus, if a token has annotations from multiple semantic artifacts, it is counted each respective row, while it only gets counted once in the row of sum of annotated token. Dividing the sum of annotated token by the overall amount of token then

Table 2 Amount of token contained as classes in semantic artifacts for token sets derived by different *min_count*, sum of annotated token, overall amount of token, and rate of annotated token

	min_count					
	1	5	10	25	50	100
AFO	218	130	97	62	42	27
BAO	100	56	37	25	15	9
CHEBI	107	42	27	23	16	5
CHMO	57	30	21	9	7	3
SBO	37	29	24	21	19	10
IUPAC-Goldbook	365	194	145	94	60	37
NCIT	935	440	300	172	103	54
Sum of annotated token	1178	537	364	211	125	65
Overall amount of token	4170	861	525	276	153	74
Rate of annotated token (in %)	28.25	62.37	69.33	76.45	81.70	87.84

yields the rate of annotated token. A high rate of annotated token is desired in order to reduce later workload in revising the ontology, as coming up with definitions for classes is more difficult than agreeing on an already existing one. However, a high sum of annotated token also is desired as integrating more classes into an ontology results in a higher expressivity of the latter.

While sets obtained by setting a low *min_count* contain more token than those with higher *min_count*, the rate of annotated token rises with higher *min_count* parameters. This also might indicate a higher relevance of the token contained in the sets with high *min_count* parameters. In addition, the rate of annotated token for a $min\_count = 1$ is quite low with 28.25 % compared to the other rates. This might be due to the inclusion of typing mistakes and non-domain relevant token at lower *min_count*, as one occurrence would suffice for the token to be contained in the text dataset. On the other hand, lower *min_count* parameters take into account more concepts not yet defined in the ontologies. These concepts in turn allow for generation of more new candidates of classes in the respective ontologies. The ontologies themselves have lower amounts of token contained compared to the thesauri. However, the AFO is expected to be the ontology best fitting to the dataset as it has the highest number of annotated token while not having the highest amount of classes compared to the other ontologies. This indicates an intersection of topics represented in the text dataset and the AFO.

Plotting the rate of annotated token against the *min_count* parameters, as in Fig. 5, the largest jump in the rate occurs between $min\_count = 1$ and $min\_count = 2$.

Taking into account the number of token found in each ontology, the AFO contains the most token for each *min_count*. Thus, the AFO is deemed as most fitting ontology of the five ontologies for the description of the knowledge domain contained in the text dataset and accordingly chosen as ontology to be extended by the method elucidated in Sect. 3.3.

Word2Vec models are trained on token sets based on *min_count* parameters in the range $min\_count = [1...25]$. Then, class labels from the AFO that are also contained in the token set are used as input to determine the most similar words. As the similarity of the words is determined by the cosine similarity, thresholds can be set to confine the amount of output words with regards to their similarity to the input word. A maximum amount of five output

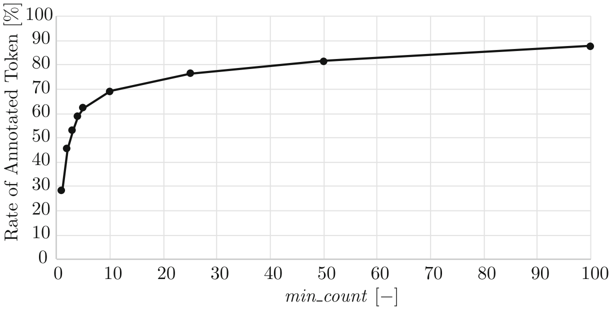


Fig. 5 Rate of annotated token for different *min_count*

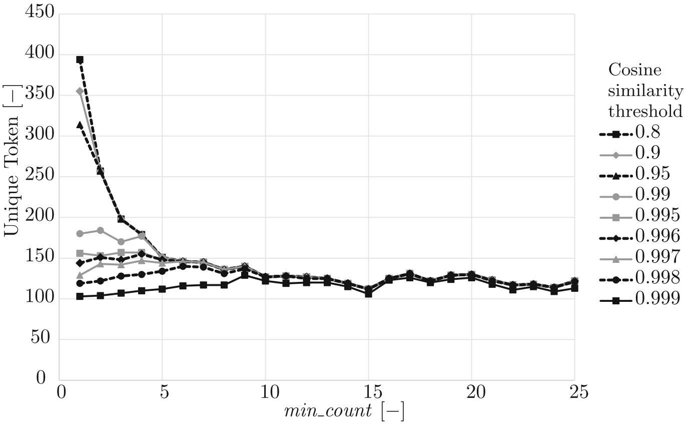


Fig. 6 Amount of unique token output by Word2Vec for classes of AFO with different *min_count* and cosine similarity thresholds varied between [0.8, ..., 0.999]

words per input word is set, and the threshold varied in the range of [0.8, ..., 0.999]. As some words are contained in multiple output sets for different input words, the amount of unique token generated by Word2Vec is calculated by only counting each word generated as a class candidate of the ontology once. With the AFO as ontology to be extended, Fig. 6 shows the amount of unique token found for different *min_count* parameters and different cosine similarity thresholds.

While the cosine similarity threshold has an impact on the amount of unique token generated for low *min_count*, the effect seems to be mitigated for thresholds in the range [0.8, ..., 0.995] and *min_count* > 5. Using different *min_count* and a cosine similarity threshold of 0.999, the AFO is extended automatically by new classes suggested by the Word2Vec model. The new classes are furthermore annotated by respective textual definitions obtained from the classes and concepts of the other semantic artifacts presented in Sect. 2.2. Object properties *conceptually related to* are asserted, pointing to the respective ontology classes already contained in the AFO before extension.

Table 3 lists the resulting number of new classes inserted into the AFO obtained by setting the cosine similarity threshold to 0.999 and applying different *min_count* parameters in the range [1, ..., 25]. In addition, the amount of annotated new classes is listed along with the number of textual definitions according to the source of the textual definition related to the corresponding semantic artifact. Here, a *min_count* of 10 seems to be the most promising one,

Table 3 Number of new classes and annotated new classes in AFO created by the workflow along with the number of annotations obtained from each respective semantic artifact

min_count	1	2	5	10	25
BAO	5	6	5	6	4
CHEBI	9	10	7	7	6
CHMO	4	3	3	3	2
SBO	6	9	6	9	7
IUPAC-Goldbook	28	24	26	28	29
NCIT	50	51	50	58	56
Annotated new classes	59	60	62	68	66
New classes	73	73	77	91	87

Extension of ontology conducted with cosine similarity threshold set to 0.999 and different *min_count*

as the number of new classes (91) and number of annotated new classes (68) are highest. Thus, the AFO is extended by 91 classes which are created automatically based on the text dataset. From these new classes, 68 are annotated based on the other semantic artifacts achieving an annotation rate of $68/91 = 74.73\%$. Of these 68 annotated new classes, 6 are annotated based on BAO class-definitions, 7 based on CHEBI, 3 based on CHMO, and 9 based on SBO classes. Furthermore, 28 classes are annotated based on IUPAC-Goldbook concepts and 58 based on the NCIT. The sum of these annotations is greater than 68, indicating multiple annotations for some new classes in the extended AFO.

The automatically added classes are concepts taken from the text dataset; thus, they may be used to describe the context represented in the 28 scientific texts. Furthermore, the semantic artifacts chosen in this publication all deal somehow with the domain of chemistry or at least are situated in the domain of natural sciences that deal with chemical substances. Thus, the annotation of the classes is assumed to be in the correct domain as the source of the annotation already is situated close to the needed domain of knowledge. As the annotations often only vary in small details, the decision on (re-)use of specific ontology classes should be done by domain experts.

To provide an example of the resulting extension, Protégé is used for visualization of the resulting ontology. Figure 7 shows the class hierarchy of the already contained AFO classes *concentration* and *rate* using blue arrows for the hierarchical relation *has subclass*.

The new class *flow* is inserted based on the workflow as subclass of *w2vConcept* and gets assigned the relation of *conceptually related to* (denoted by dashed orange arrows) connecting it to the classes *concentration* and *rate*.

Furthermore, the new class *flow* gets annotated by the textual definition of the concept *flow* found in the NCIT. The resulting annotations of the class *flow* are depicted in Fig. 8. The first entry contains the label of the class, while the next two entries point to the word-input that led to the generation of the class. The bottommost entry contains a textual definition found in the NCIT. The remark ‘Found in [NCIT]’ gives the link to the underlying class of the ontology, allowing for later reuse of the respective entity. As the new classes are generated automatically, an arbitrary amount of such *rdfs:comment* can be assigned to a class, but only one *rdfs:label* is assigned.

Thus, an existing ontology can be extended automatically by concepts based on scientific texts. After extension of the ontology, an evaluation by domain experts should be conducted, as not every resulting definition and relation might be correct.

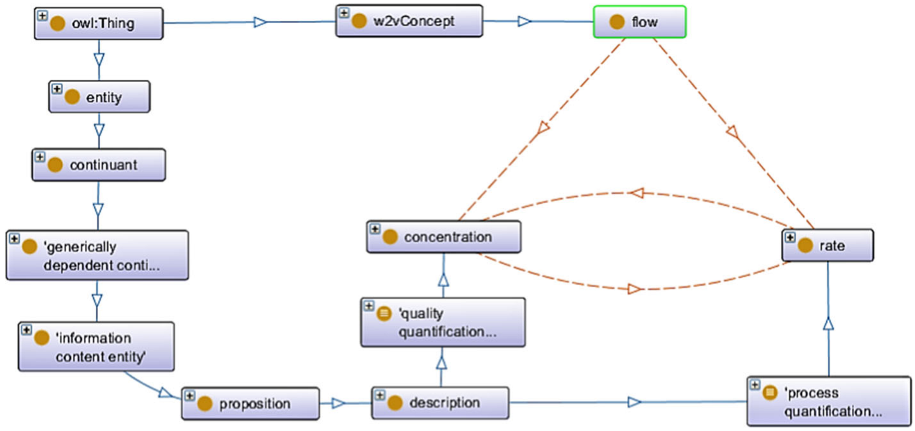
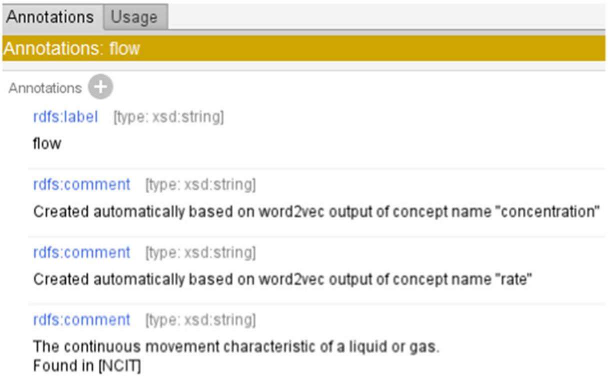

owl:Thing
w2vConcept
flow
entity
continuant
'generically dependent conti...
concentration
rate
'quality quantification...
'information content entity'
proposition
description
'process quantification...

Fig. 7 Visualization of class hierarchy of new class *flow* in Protégé. Class *flow* and relations *conceptually related to* to existing classes created automatically by the workflow with *min_count* = 10 and cosine similarity threshold = 0.999. Solid blue arrows indicate relation *has subclass*, dashed orange arrows denote relation *conceptually related to*



rdfs:label [type: xsd:string]
flow

rdfs:comment [type: xsd:string]
Created automatically based on word2vec output of concept name "concentration"

rdfs:comment [type: xsd:string]
Created automatically based on word2vec output of concept name "rate"

rdfs:comment [type: xsd:string]
The continuous movement characteristic of a liquid or gas.
Found in [NCIT]

Fig. 8 Annotations of new class *flow* visualized in Protégé for later review by domain experts

This in turn can be used for an automated, ontology aligned annotation of research data: When a researcher uploads their research data and corresponding textual documentation to a database, the workflow presented in this work can then be used to automatically choose the best fitting ontology and extend it. The extended ontology could then in turn be used to annotate the previous uploaded research data, linking data entries with relations as posed in the textual documentation.

5 Summary and outlook

5.1 Conclusion

Ontologies are used to describe knowledge in an explicit and machine-readable way, while still being human-readable. Thus, they are used to model knowledge and semantic relations between data and concepts of scientific knowledge domains.

In this contribution, a method is set up to automatically make use of natural language processing (NLP) techniques to extract concepts contained in a text dataset in order to extend existing ontologies by these concepts relevant to a domain of knowledge. A search for textual concept definitions from different sources such as different ontologies and thesauri allows for automated annotation of these concepts found. This also helps in picking the right ontology to be extended in the second part of the workflow, where the extension of an ontology is performed by new classes based on the text dataset. Different word vectorization models using Word2Vec are trained based on different allowed numbers of repetitions of the token within the preprocessed text dataset (*min_count*) and used to suggest new classes and relations between them. Finally, the classes are annotated with textual definition based on other ontologies and thesauri, where possible.

This workflow allows for automated extension of ontologies by classes contained as concepts in a text dataset. A text dataset of 28 papers on the topic of catalytic methanation of CO_2 reactions, five ontologies and two thesauri are used as a proof-of-concept. While use of a low *min_count* parameter results in higher numbers of new classes suggested, it also allows for integration of concepts not that important to the domain of knowledge, as the lower rates of annotated token suggest. Using a *min_count* parameter of 10, the Allotrope Foundation Ontology (AFO) is extended automatically by 91 new classes obtained by the text dataset. Of these classes, 68 classes are provided automatically with at least one textual definition based on the other semantic artifacts (i.e., the other ontologies and thesauri) provided.

This workflow can easily be adapted for other ontologies and text datasets to extend existing ontologies. Additionally, the database of semantic artifacts can be set for a larger number of ontologies and thesauri. While this can be adjusted quickly, the use of other definition databases such as WikiData can be implemented with some code adjustments.

5.2 Limitations and future work

The workflow only uses single-word tokens, thus only is able to search for and add single-word classes to the ontology. Detecting multi-word concepts with the presented workflow is not yet possible, but desirable as often ontology classes consist of more than one word. In the future, manipulation of the applied POS-tagging is planned to mitigate the limitation of only single-word classes being considered by the presented workflow. Here, e.g., neighboring noun token could be combined to one class, such as “flow rate”, or pairs of neighboring adjective and noun pairs, like “catalytic reaction.” Furthermore, the use of more sophisticated methods, such as named entity recognition (NER) [28], can be used. However, this method requires the pre-definition of categories. While this is already quite available for general categories, the definition of catalysis-related categories for NER is yet to be implemented to the best knowledge of the authors.

The second major limitation of the presented workflow is the missing refinement of the “semantically related to” relation used to link existing and newly created classes. The relationships could not be further refined because the semantic relation of the concepts is not appropriately given by word2vec. For example, no distinction is made between a hierarchical relationship or an object property. This is also due to the fact that only nouns are included as classes into the ontology; thus, verbs and adjectives are not considered, which would be the more fitting candidates for ontology properties and relations. In future work, relationship extraction and entity linking, i.e., the Radbound Entity Linker [29] could be used to develop more sophisticated relationship extraction. After extracting the relations, additional linking to already existing ontology relationships is also in the scope.

To evaluate the usefulness of the workflow, an evaluation by domain experts should be conducted, to classify the number of valuable classes and relations generated automatically by the workflow. Extending an ontology by textual input as shown in this work also will help domain experts in the future to automatically annotate research data when uploading a set of research data together with a corresponding paper to a research database.

Supplementary information

The code developed in this work is available in a GitHub repository here: https://github.com/TUDoAD/NLP-Based-Ontology-Extender.

The pre-processed pdf-files and the ontology files are available in a zenodo repository here: https://zenodo.org/record/7956870.

Acknowledgements The authors thank the Deutsche Forschungsgemeinschaft (DFG) within the Nationale Forschungsdateninfrastruktur (NFDI) initiative (Grant No.: NFDI/2-1-2021) for funding part of this work as well as the fruitful discussions with researchers in NFDI4Cat. A.S.B. thanks the networking program “Sustainable Chemical Synthesis 2.0” (SusChemSys 2.0) for the support and fruitful discussions across disciplines.

Funding Open Access funding enabled and organized by Projekt DEAL. The Deutsche Forschungsgemeinschaft (Grant No.: NFDI/2-1-2021) funded part of this work.

Declarations

Conflict of interest The authors declare no competing interests as defined by Springer, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

Appendix A: References of text dataset

See Table 4.

Table 4 Listing of the references used as text dataset of 28 Papers on Methanation of CO_2

No	Author	Title
1	Jie Liu et al.	Alkaline-assisted Ni nanocatalysts with largely enhanced low-temperature Activity toward CO2 methanation https://doi.org/10.1039/c5cy02026c
2	Karim Ghaib et al.	Chemical Methanation of CO2: a review https://doi.org/10.1002/cite.201600066
3	Kechao Zhao et al.	CO2 methanation and co-methanation of CO and CO2 over Mn-promoted Ni/Al2O3 catalysts https://doi.org/10.1007/s11705-016-1563-5
4	Bo Liu, Shengfu Liu	Comparative study of fluidized-bed and fixed bed reactor for syngas methanation over Ni-W/TiO2-SiO2 catalyst https://doi.org/10.1016/S2095-4956(13)60098-4
5	Carlos V. et al.	Direct CO2 hydrogenation to methane or methanol from post-combustion Exhaust streams—a thermodynamic study https://doi.org/10.1016/j.jngse.2014.11.010
6	Fabian Grueger et al.	Early power to gas applications: reducing wind farm forecast errors and providing Secondary control reserve https://doi.org/10.1016/j.apenergy.2016.06.131
7	T.T.M. Nguyen et al.	High temperature methanation: catalyst considerations https://doi.org/10.1016/j.cattod.2013.03.035
8	Bin Lu et al.	Highly Dispersed Ni Nanocatalysts Derived from NiMnAl-Hydrotalcites as High-Performing Catalyst for low temperature Syngas Methanation https://doi.org/10.3390/catal9030282
9	O. Görke et al.	Highly Selective methanation by the use of microchannel reactor https://doi.org/10.1016/j.cattod.2005.09.009
10	Jonathan Lefebvre et al.	Improvement of three-phase methanation reactor performance for steady-state and transient operation https://doi.org/10.1016/j.fuproc.2014.10.040

Table 4 continued

No	Author	Title
11	Claudia Krier et al.	Improving the Methanation Process https://doi.org/10.1002/cite.201200221
12	Qiushi Pan et al.	Insight into the reaction route of CO2 methanation: Promotion effect of medium basic sites https://doi.org/10.1016/j.catcom.2013.10.034
13	Ben Redondo et al.	Intensified isothermal reactor for methanol synthesis https://doi.org/10.1016/j.cep.2019.107606
14	Maria C. Bacariza et al.	Magnesium as Promoter of CO2 Methanation on Ni-Based USY Zeolites https://doi.org/10.1021/acs.energyfuels.7b01553
15	Wie Wang, Jinlong Gong	Methanation of Carbon dioxide: an overview https://doi.org/10.1007/s11705-010-0528-3
16	Kriston P. Brooks et al.	Methanation of carbon dioxide by hydrogen reduction using the Sabatier process in microchannel reactors https://doi.org/10.1016/j.ces.2006.11.020
17	Antoine Beuls et al.	Methanation of CO2: Further insight into the mechanism over Rh/γ-Al2O3 catalyst https://doi.org/10.1016/j.apcatb.2011.02.033
18	Athanasia Petala et al.	Methanation of CO2 over alkali-promoted Ru/TiO2 catalysts: I. Effect of alkali additives on Catalytic activity and selectivity https://doi.org/10.1016/j.apcatb.2017.11.048
19	Chuanfei Liang et al.	Methanation of CO2 over Ni/Al2O3 modified with alkaline earth metals: Impacts of oxygen Vacancies on catalytic activity https://doi.org/10.1016/j.ijhydene.2019.02.014
20	Axel Fache et al.	Optimization of fixed-bed methanation reactors: safe and efficient operation under transient And steady-state conditions https://doi.org/10.1016/j.ces.2018.08.044
21	Jia Zhang et al.	Preparation of graphene oxide-based surface plasmon resonance biosensor with Au Bipyramid nanoparticles as sensitivity enhancer https://doi.org/10.1016/j.colsurfb.2014.01.003

Table 4 continued

No	Author	Title
22	Muhammed Younas et al.	Recent Advancements, Fundamental Challenges, and Opportunities in Catalytic Methanation of CO2 https://doi.org/10.1021/acs.energyfuels.6b01723
23	Jiajian Gao et al.	Recent advances in methanation catalysts for the production of synthetic natural gas https://doi.org/10.1039/C4RA16114A
24	Woo Jin Lee et al.	Recent trend in thermal catalytic low temperature CO2 methanation: A critical review https://doi.org/10.1016/j.cattod.2020.02.017
25	Robert A. Dagle et al.	Selective CO methanation catalysts for fuel processing applications https://doi.org/10.1016/j.apcata.2007.04.015
26	Waqar Ahmad et al.	Synthesis of lanthanide series promoted Ni/γ-Al2O3 catalysts for Methanation of CO2 at low temperature under atmospheric pressure https://doi.org/10.1016/j.catcom.2017.06.044
27	Duo Sun, David Simakov	Thermal management of a Sabatier reaction for CO2 conversion into CH4: simulation-based analysis https://doi.org/10.1016/j.jcou.2017.07.015
28	Martin P. Andersson et al.	Toward computational screening in heterogeneous catalysis: Pareto-optimal methanation catalysts https://doi.org/10.1016/j.jcat.2006.02.016

References

1. Wulf C, Beller M, Boenisch T, Deutschmann O, Hanf S, Kockmann N, Kraehnert R, Oezaslan M, Palkovits S, Schimmler S, Schunk SA, Wagemann K, Linke D (2021) A unified research data infrastructure for catalysis research-challenges and concepts. ChemCatChem 13(14):3223–3236. https://doi.org/10.1002/cctc.202001974
2. Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, Blomberg N, Boiten J-W, da Silva Santos LB, Bourne PE, Bouwman J, Brookes AJ, Clark T, Crosas M, Dillo I, Dumon O, Edmunds S, Evelo CT, Finkers R, Gonzalez-Beltran A, Gray AJG, Groth P, Goble C, Grethe JS, Heringa J, 't Hoen PAC, Hooft R, Kuhn T, Kok R, Kok J, Lusher SJ, Martone ME, Mons A, Packer AL, Persson B, Rocca-Serra P, Roos M, van Schaik R, Sansone S-A, Schultes E, Sengstag T, Slater T, Strawn G, Swertz MA, Thompson M, van der Lei J, van Mulligen E, Velterop J, Waagmeester A, Wittenburg P, Wolstencroft K, Zhao J, Mons B (2016) The fair guiding principles for scientific data management and stewardship. Sci Data 3(1):160018. https://doi.org/10.1038/sdata.2016.18
3. Strömert P, Hunold J, Castro A, Neumann S, Koepler O (2022) Ontologies4chem: the landscape of ontologies in chemistry. Pure Appl Chem 94(6):605–622. https://doi.org/10.1515/pac-2021-2007
4. Gruber TR (1993) A translation approach to portable ontology specifications. Knowl Acquis 5(2):199–220. https://doi.org/10.1006/knac.1993.1008
5. National Cancer Institue: National Cancer Institue Thesaurus. https://ncit.nci.nih.gov (2022)
6. Grühn J, Behr AS, Eroglu TH, Trögel V, Rosenthal K, Kockmann N (2022) From coiled flow inverter to stirred tank reactor—bioprocess development and ontology design. Chem Ing Tec 94(6):852–863. https://doi.org/10.1002/cite.202100177
7. Menke MJ, Behr AS, Rosenthal K, Linke D, Kockmann N, Bornscheuer UT, Dörr M (2022) Development of an ontology for biocatalysis. Chemie Ingenieur Technik 94(11):1827–1835. https://doi.org/10.1002/cite.202200066
8. Asim MN, Wasim M, Khan MUG, Mahmood W, Abbasi HM (2018) A survey of ontology learning techniques and applications. Database. https://doi.org/10.1093/database/bay101
9. Dal A, Maria J (2012) Simple method for ontology automatic extraction from documents. Int J Adv Comput Sci Appl. https://doi.org/10.14569/ijacsa.2012.031206
10. Opasjumruskit K, Peters D, Schindler S (2020) DSAT: Ontology-based information extraction on technical data sheets. ISWC 2020, 2–6, Nov. 2020. https://ceur-ws.org/Vol-2721/paper563.pdf
11. Opasjumruskit K, Böning S, Schindler S, Peters D (2022) OntoHuman: ontology-based information extraction tools with human-in-the-loop interaction. In: International conference on cooperative design, visualization and engineering. Springer, Berlin, pp 68–74
12. Opasjumruskit K (2020) NLP for ontology development-a use case in spacecraft design domain. https://elib.dlr.de/136233/
13. Horsch M, Petrenko T, Kushnarenko V, Schembera B, Wentzel B, Behr A, Kockmann N, Schimmler S, Bönisch T (2022) Interoperability and architecture requirements analysis and metadata standardization for a research data infrastructure in catalysis. In: Pozanenko A, Stupnikov S, Thalheim B, Mendez E, Kiselyova N (eds) Data analytics and management in data intensive domains. Springer, Cham, pp 166–177. https://doi.org/10.1007/978-3-031-12285-9_10
14. NFDI4Cat: Ontology collection of NFDI4Cat. https://nfdi4cat.org/en/services/ontology-collection (2022)
15. Allotrope Foundation: Allotrope Foundation Ontology. https://www.allotrope.org/ontologies (2022)
16. Hastings J, Owen G, Dekker A, Ennis M, Kale N, Muthukrishnan V, Turner S, Swainston N, Mendes P, Steinbeck C (2015) ChEBI in 2016: improved services and an expanding collection of metabolites. Nucleic Acids Res 44(D1):1214–9
17. Batchelor C (2022) Chemical methods ontology. http://purl.obolibrary.org/obo/chmo.owl
18. Abeyruwan S, Vempati UD, Küçük-McGinty H, Visser U, Koleti A, Mir A, Sakurai K, Chung C, Bittker JA, Clemons PA, Brudz S, Siripala A, Morales AJ, Romacker M, Twomey D, Bureeva S, Lemmon V, Schürer SC (2014) Evolving BioAssay ontology (BAO): modularization, integration and applications. J Biomed Semant. https://doi.org/10.1186/2041-1480-5-s1-s5
19. Nguen T, Karr J, Sheriff R (2022) Systems biology ontology. http://biomodels.net/SBO/
20. Gold V (ed.) (2019) The IUPAC compendium of chemical terminology. International Union of Pure and Applied Chemistry (IUPAC). https://doi.org/10.1351/goldbook
21. Musen MA (2015) The protégé project: a look back and a look forward. AI Matters 1(4):4–12. https://doi.org/10.1145/2757001.2757003
22. Řehůřek R, Sojka P (2010) Software framework for topic modelling with large corpora. pp 45–50. https://doi.org/10.13140/2.1.2393.1847

23. Pennington J, Socher R, Manning C.D (2014) Glove: global vectors for word representation. In: Empirical methods in natural language processing (EMNLP), pp 1532–1543. http://www.aclweb.org/anthology/D14-1162
24. Adobe Inc (2022) Adobe Acrobat Pro PDF-reader, version 22.003.20258. https://www.adobe.com/acrobat.html
25. Shinyama Y (2007) PDFMiner—Python PDF Parser. https://github.com/euske/pdfminer
26. Honnibal M, Montani I, Van Landeghem S, Boyd A (2020) spaCy: industrial-strength natural language processing in Python. https://doi.org/10.5281/zenodo.1212303
27. Lamy J-B (2017) Owlready: ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies. Artif Intell Med 80:11–28. https://doi.org/10.1016/j.artmed.2017.07.002
28. Nadeau D, Sekine S (2007) Named entities: recognition, classification and use. Lingvist Investig 30(1):3–26. https://doi.org/10.1075/li.30.1.03nad
29. van Hulst JM, Hasibi F, Dercksen K, Balog K, de Vries AP (2020) Rel: an entity linker standing on the shoulders of giants. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. SIGIR'20. ACM

Alexander S. Behr is a PhD-Student at the Laboratory of Equipment Design of Norbert Kockmann since 2020, working on ontology development in catalysis research within the NFDI4Cat Project. He studied Chemical Engineering at TU Dortmund University, where he achieved his Master in Chemical Engineering from TU Dortmund University in the domain of Computational Fluid Dynamics (CFD).

Marc Völkenrath is currently doing his master's thesis at the Laboratory of Equipment Design. He did research in the field of ontologies and natural language processing as part of his bachelor's thesis in 2022 and subsequently worked as research assistant.

Norbert Kockmann is full professor at TU Dortmund University since 2011. He achieved his diploma from TU Munich in Mechanical Engineering and his doctorate at Bremen University in Technical Thermodynamics. He worked for several years in various positions in process industry. At Freiburg University, he achieved his habilitation in Microsystems Engineering in 2007. His research interests are on modular small-scale devices for continuous flow process intensification. His research includes multiphase flow with experimental and simulation work assisted by machine-learning methods.