Joint Research Programme
BTO 2024.012 | January 2024

Zeer zorgwekkende stoffen (deel 3) –

Literature mining

Joint Research Programme

Bridging Science to Practice

Colophon

Zeer zorgwekkende stoffen in het milieu (deel 3) – Literature mining

BTO 2024.012| January 2024

This research is part of the Joint Research Programme of KWR, the water utilities and Vewin.

Project number
402045/233

Project manager
Dr. Patrick Bäuerlein

Client
BTO - Thematical research - Hydroinformatics

Author(s)
Dr. Xin Tian, Siddharth Seshan, Dr. Bas Wols

Quality Assurance
Dr. Peter van Thienen

Sent to
This report is distributed to BTO-participants.

A year after publication it is public.

Keywords

Literature mining, HydroInformatics, Cheminformatics, Natural Language Processing, Generative Pre-trained Transformer (GPT)

Year of publishing
2024

More information
Xin Tian
T +31 615184269
E xin.tian@kwrwater.nl

PO Box 1072
3430 BB Nieuwegein
The Netherlands

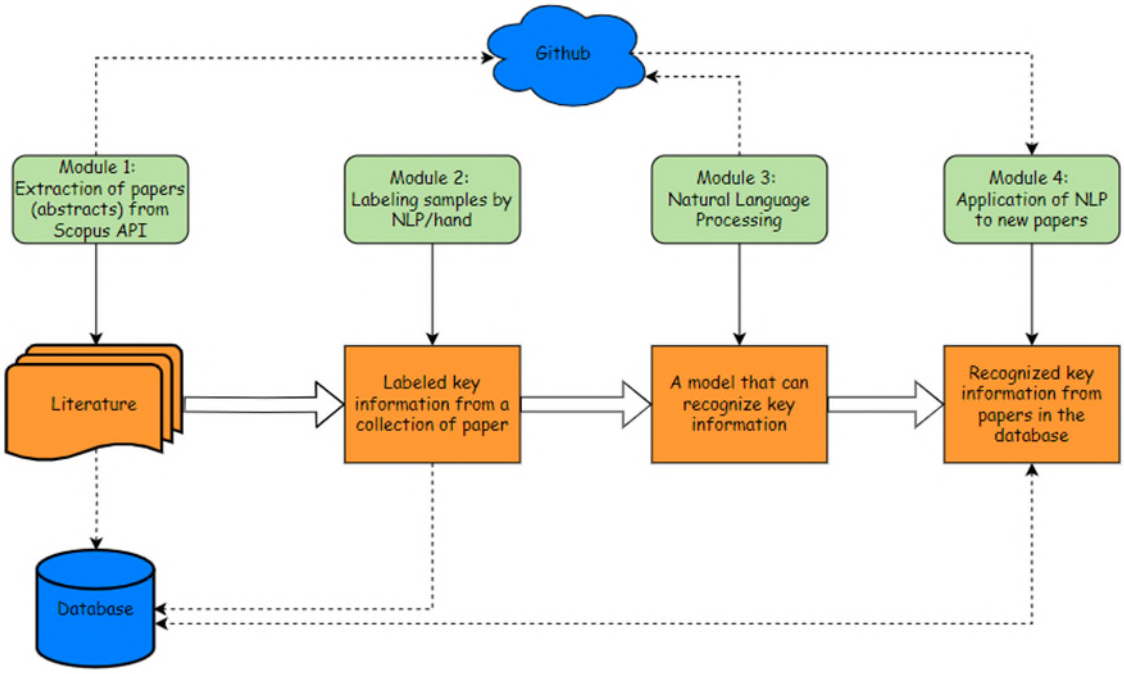
T +31 (0)30 60 69 511
E info@kwrwater.nl
I www.kwrwater.nl

Managementsamenvatting

Een workflow om wateronderzoek te versterken met Natural Language Processing (NLP) en Large Language Models (LLM)

Auteur Xin Tian, Siddharth Seshan, Bas Wols

Natural Language Processing (NLP) en Large Language Models (LLM) kunnen een cruciale rol vervullen bij literatuuronderzoek door het analyseren van geschreven teksten te automatiseren, belangrijke concepten te identificeren en relevante informatie te extraheren. Er is een benadering ontwikkeld voor literatuuronderzoek met behulp van NLP en LLM's door NLP in te zetten bij het zoeken naar informatie over chemische verbindingen. Dit rapport beschrijft de methode die wordt gebruikt voor geautomatiseerd downloaden van artikelen en informatie-extractie met behulp van NLP, weergegeven in onderstaande figuur.



Workflow om het zoeken en verwerken van wetenschappelijke informatie te automatiseren

Belang: grote hoeveelheden informatie uit literatuur extraheren kan onderzoek versterken

De huidige Natural Language Processing (NLP) en Large Language Models (LLM) zijn in staat om de analyse van geschreven teksten te automatiseren, daarbij belangrijke concepten te identificeren en de relevante informatie te extraheren. NLP-technieken worden gebruikt bij tekstclassificatie en het herkennen van benoemde entiteiten (de zogenoemde *name entity recognition* of NER). De integratie van Generative Pre-trained Transformer (GPT)-modellen, zoals GPT-3.5 of GPT-4, heeft het literatuuronderzoek verder verbeterd door de analyse van grote hoeveelheden teksten mogelijk te

maken en automatische samenvattingen te genereren. Dit kan ook een grote waarde hebben voor wateronderzoek. Zo kan relevante informatie over reactiesnelheden en kwantumrendementen extraheren uit een grote hoeveelheid chemische informatie in de literatuur inzicht bieden in de kinetiek van chemische processen. Dat vergroot bijvoorbeeld de efficiëntie van onderzoek naar de waterkwaliteit en -kwantiteit, en ondersteunt onderzoek op uiteenlopende gebieden als waterzuivering en geohydrologie. Inzet van LLM en NLP maakt trendanalyse, modelontwikkeling en het creëren van uitgebreide databases mogelijk en kan onderzoekers helpen lacunes in kennis aan te pakken, nieuwe oplossingsrichtingen voorstellen en samenwerking stimuleren. Het is daarom belangrijk om te onderzoeken hoe LLM en NLP het beste kunnen worden ingezet in wateronderzoek.

Aanpak: NLP en taalmodellen ingezet om relevante chemische informatie uit artikelen te halen

Er is geautomatiseerd gezocht naar artikelen om te downloaden op basis van auteursnamen en zoekwoorden met behulp van de Scopus API. Twee taalmodellen, ontwikkeld door KWR en OpenAI, werden getest om in de literatuurnformatie te vinden over chemische verbindingen en hun reactiesnelheidsconstanten. Deze gegevens zijn van belang voor waterbehandelingsstudies naar de afbraak van stoffen met geavanceerde oxidatie.

Resultaten: NLP is toepasbaar maar arbeidsintensief, GPT werkt goed en belooft meer

Het is mogelijk NLP modellen zelf te trainen, maar dat is arbeidsintensief omdat daarvoor veel domeinspecifieke voorbeelden moeten worden gemaakt. Direct gebruik maken LLM's, zoals het GPT 3.5 Turbo-model, werkte beter. Met deze aanpak kon relevante numerieke informatie automatisch worden gevonden, zoals werd bevestigd voor een toepassing met fotochemische afbraakconstanten voor waterbehandelingsprocessen. Hierdoor kon een bestaande database met stofgegevens aanzienlijk worden vergroot.

Toepassing: inzet LLM en NLP maakt nauwkeurigere analyses en onderzoeken mogelijk

Door LLM’s in te zetten kunnen onderzoekers en professionals van KWR, waterlabs en drinkwaterbedrijven relevante informatie halen uit grote hoeveelheden tekst, zoals informatie over chemische reacties, PFAS, microplastics of natuurlijke gevaren. Dit kan hen op een snelle manier helpen om op de hoogte te blijven van de nieuwste bevindingen en de reikwijdte van hun onderzoek te vergroten, en maakt het mogelijk om grotere hoeveelheden numerieke en tekstuele informatie uit de literatuur te verzamelen. Dit maakt uitgebreidere en nauwkeurigere analyses en onderzoeken mogelijk.

Rapport

Dit onderzoek is beschreven in het rapport *Zeer zorgwekkende stoffen in het milieu (deel 3) – Literature mining* (BTO-2024.012). Lees meer over ZZS in de BTO rapporten:

- Zeer zorgwekkende stoffen (deel 1) - clustering, bemonstering en toxiciteit, BTO 2024.010 [8].
- Zeer zorgwekkende stoffen (deel 2) – zuivering, BTO 2024. 011 [9].
- Verslag van veldmetingen en historische meetgegevens over afbraak van organische microverontreinigingen in grondwater, BTO2024.013 [10].

Contents

1 Introduction

1.1 Natural Language Processing and literature mining

Natural Language Processing (NLP) is a subfield of computer science and artificial intelligence that deals with the interaction between machines and human languages. It enables machines to try to understand, interpret, and generate human language, making it a powerful tool for various applications. One of the areas where NLP has shown potential is in Literature Mining, which typically is the automated process of extracting relevant and valuable information from large volumes of written texts. It involves analyzing the structure and content of written texts to identify named entities and relationships of information.

NLP is a key component of automated text interpretation. By using NLP techniques, machines can analyze written texts, identify key concepts, and extract relevant information. One of the primary applications of NLP in Literature Mining is in text classification, which involves categorizing written texts into different categories based on their content. Another application of NLP in Literature Mining is Named Entity Recognition (NER), which involves identifying and categorizing information into pre-defined classes, e.g., locations of interest, names of chemical components.

Recently, there have been significant advances in NLP technology, particularly with the development of Generative Pre-trained Transformer (GPT) models. These models are capable of interpreting and generating text at a near-human level, making them potentially useful in Literature Mining. For example, GPT can potentially be used to analyze large volumes of texts, find information, and generate summaries automatically, which can save researchers a significant amount of time and effort. As such, the integration of GPT technology into Literature Mining has the potential to further advance many fields.

1.2 Application in searching information of chemical compounds

Therefore, this project aimed to demonstrate the value of LLMs for literature mining through a case study. We applied NLP methods to broadly search information about rate constants and quantum yields of chemical compounds for UV based water treatment processes. They are used in kinetic modeling, which is important for determining the removal rate of a chemical substance. Rate constants determine the rate of a chemical compound with a specific chemical that is used in the water treatment process (for example ozone). Quantum yield, on the other hand, is a measure of the efficiency of a photochemical process. It is used to evaluate the effectiveness of photoreactions and photochemical processes. In our case study, we aim to employ NLP to search for rate constants and quantum yield information from scientific papers, where information is often described in an unstructured way.

1.3 Approach

In this project, we first conducted automated paper searching and downloading based on key words or author names. Using their abstracts, we tested two models, one small-scale and domain-specific language model developed by KWR and one large-scale generic language model developed by OpenAI, to find information about chemical compounds and their rate constant values. Models results were compared to provide suggestions for readers to consider the use of language models for literature mining in other situations.

1.4 Implication of this study for other work packages of this project

This report was conducted as part of the BTO project 'Zeer zorgwekkende stoffen (ZZS) in het milieu' (substances of very high concern). This is an overarching BTO project of the BTO themes "Bronnen & Omgeving", "Chemische Veiligheid", "Zuivering" and "Hydroinformatics". A number of reports have been published on the study of Substances of Very High Concern. The first sub-report describes the clustering, sample program and toxicity of ZZS. The water treatment aspects of ZZS are described in [8]. And a report on the behavior of ZZS in the subsurface has been published [9]. The current report describes the method of literature mining and the results were used in the water treatment work package of the project (BTO 2024.011) and will continue to be useful in the future for other projects.

1.5 Structure of this report

This report first describes a method developed to automate paper downloading based on keywords or author names in Section 2. Specifically in the case study, "rate constant" and "quantum yield" were used as the keywords to search for paper abstracts (around 5000). Section 3 introduces a language model developed by KWR which can be used to find information. Then a large language model, developed by OpenAI, was also used to carry out the same task (Section 4). Conclusions and recommendations follow in Section 5.

2 Automation of literature downloading and processing

2.1 Motivation and background information

Researchers and practitioners are always on the lookout for ways to improve the efficiency of their work. One area that has seen a considerable amount of innovation in recent years is the use of application programming interfaces (APIs) to automate the process of downloading research papers and their abstracts. For example, Scopus, a comprehensive scientific database, offers an API that allows researchers to access and download open-access papers or abstracts of all searchable papers [1].

We first used the Scopus API to download abstracts automatically by considering several advantages of using it. First, it can save researchers a considerable amount of time that would otherwise have been spent manually downloading them. With the API, one can automate the process of downloading abstracts, which can help them to focus on other critical aspects of their research. Second, using Scopus API ensures the accuracy of the abstracts that researchers download. The API pulls abstracts directly from the Scopus database, which is updated continuously. Lastly, the use of Scopus' API can help researchers to broaden the scope of their research. The API allows researchers to search for abstracts based on specific keywords, author names, and other information. This feature can help researchers to identify new and relevant research papers that may have otherwise gone unnoticed.

Apart from Scopus, readers can also consider the use of other databases for scientific publications, for example [2]. It should be noted that Scopus is a comprehensive database which comprise most scientific papers and its API is easy to use. Domain-specific databases, such as PubMed for medical and chemical research, can also be considered while conducting relevant studies.

2.2 Scopus API

Scopus is a comprehensive scientific database owned by Elsevier that provides access to over 76 million records across various scientific disciplines. Scopus API is a tool that allows developers to access and use the data stored in the Scopus database in various applications. The API provides access to metadata, abstracts, and citation data for various scientific publications, including journals, books, conference proceedings, and patents. Developers can also use the API to search for specific topics, authors, and publications, and retrieve data in various formats, including JSON and XML.

To use the Scopus API, developers need to register for an API key and follow the API documentation to make requests. Pyscopus is the main Python wrapper we used in this project for searching and downloading paper abstracts. Readers can find more information about it from [3].

2.3 Automated downloading of papers

In WP3 of this project, we have developed a python module that calls Scopus' API to download abstracts based on given keywords or author names (Figure 1). In addition, references of all the found papers can further be processed and downloaded. Particularly in this project, we used key words – 'rate constant' and 'quantum yield' – for literature

searching. The found papers are stored in a database with the following properties (meta-data): 'scopus_id', 'title', 'publication_name', 'issn', 'isbn', 'eissn', 'volume', 'page_range', 'cover_date', 'doi', 'citation_count', 'affiliation', 'aggregation_type', 'subtype_description', 'authors', 'Abstract', 'keywords', and 'reference' (Figure 2). It should be noted that we can further search for papers in the citation list of the found paper. The function can be called in the following way, where we can define the upper limit of papers to be found using 'Lim_nr' and whether the references are expected to be searched as well using 'Lim_lvl'. The detailed code can be found from KWR's Github repository (https://github.com/KWR-Water/BTO402045233_ZZS, which is available upon request).

```
# define keywords
keywords = "'quantum yield' AND 'rate constant'"
# download publications
df_publication, dct_pubchem = load_pub_kwrcompounds(keywords, save_name_lit
='literature_rateconstant.pkl', reload_pub=True, Lim_nr=1000, Lim_lvl=0)
```

Figure 1 A snippet of codes used to download paper abstracts based on given keywords.

Index	scopus_id	title	publication_name	issn	isbn	eissn	volume	page_range	cover_date	doi
0	85151310445	The photolytic beha…	Journal of Hazardous Materials	03043894	None	18733336	452	None	2023-06-15	10.1016/j.jhazmat.2023.131320
1	85150840509	Autochthonous DOM h…	Journal of Hazardous Materials	03043894	None	18733336	451	None	2023-06-05	10.1016/j.jhazmat.2023.131027
2	85151267772	Performance evaluat…	Chemosphere	00456535	None	18791298	327	None	2023-06-01	10.1016/j.chemosphere.2023.138540

Figure 2 Three example papers found by the API, stored in a Dataframe.

3 Key (scattered) information extraction – NER & Annotations

As an initial exploration of utilizing NLP methods for identifying key information regarding certain properties of chemical compounds available in scientific papers, the technique Named Entity Recognition (NER) was investigated for the given task. NER identifies and classifies named entities that exist in text data, which are entities or object that have names and be classified or grouped in predefined classes. In other words, a NER extracts information that is scattered in text and categorizes them into predefined and agreed upon categories. For the training and testing of a case specific NER, the data related to the scientific papers as extracted using the Scopus API Python module described in Section 2 was used. Section 3.1 provides a description of the processing of the compiled dataset. A brief explanation of the NER models and the associated tools used is described in Section 3.2, which is followed by the results obtained from the model training in Section 3.3. Finally, some concluding remarks and key learnings are discussed in Section 3.4.

3.1 Text Annotations

3.1.1 Overview

Text annotation are required to process textual datasets with an aim of preparing a training dataset used for the training or fine-tuning of NLP models, which can then be used for a specific application. Annotating of texts helps creating the ground truth data which then allows for model training in a supervised learning format and also assist in model evaluation. The annotations provide Broadly, the five common text annotations include:

- *Entity recognition*: The identification and labelling of entities in the text, which could include the tagging of entities with names, which is NER, and the annotation of functional elements within a speech, such as adjectives and nouns. The latter is known as part-of-speech (POS) tagging.

- *Entity linking*: The linking of certain entities to represent the relationship between entities and its relevance to the overall context of the text. It can also involve connecting entities within a text to larger external repositories of data about them.

- *Text classification*: The categorization and classification of text or documents to predefined categories based on the content.

- *Sentiment annotation*: The labelling of emotions, sentiments or opinions that are inherently prevalent within the body of text.

- *Linguistic annotation*: Also known as corpus annotation;, this is the tagging of grammatical and semantic elements within the text.

In this project, only *entity recognition* and *entity linking* were actively pursued in the text annotation activities pursued for dataset preparation. An existing and widely used corpus was utilized which already provided *Linguistic annotations* for the English language. *Text classification* and *sentiment annotation* were not deemed necessary for this specific NER task.

3.1.2 Text Samples and User-defined Label Descriptions

The Python module described in Section 2 was utilized for the extraction of abstracts of scientific papers of relevance. The NER task focused on the identification of specific properties and the associated experimental or environmental conditions of chemical compounds and therefore, the SCOPUS search using the Python module utilized the following two keywords, "rate constant and "quantum yield" (also see Section 2.2). In doing so, 207 raw texts from the abstracts of about 100 papers were extracted and further utilized.

To conduct the text annotation for the given task, pre-defined labels were created to categorize the entities and the dependencies between entities were created. The entity labelling ensured that labels are assigned to the specific contents within the text based on what they specifically represent. The labels used to assign to entities with an appropriate description and example are provided in Table 1.

Table 1:Entity labels used for text annotations and model training

Entity Label	Description	Example
compounds	Chemical substance in question as found in the text.	Chlortoluron
properties	Characteristic of a particular substance that can be observed in a chemical reaction.	Quantum yield, rate constant
agents	An agent (chemical substance, reagent, catalyst, material, energy) that causes the reaction of the compounds.	UV, H_2O_2, O_3
conditions	The specific condition (experimental or environmental) under which the reaction occurs.	Temperature value, pH value
values	Numeric values found in the text.	3×10^{-10}
units	Units of the numeric values found in the text.	s^{-1}

Additionally, dependencies between the entities were also annotated. This involved the specification of any relations or linkages that occur between entities previously labelled.

Table 2: Dependency labels used for text annotations and model trainings

Dependency Label	Description	Example
property-agent (PROP_AGEN)	A given chemical compound used/present in the reaction that can be associated with a given chemical property observed.	**Rate constant** of a given chemical compound as witnessed when **ozone** is used/present as an agent.
property-condition (PROP_COND)	A given condition associated to the chemical reaction that can be associated with a given chemical property observed.	**Rate constant** of a given chemical compound as witnessed when a certain **pH** is maintained.

property-value (PROP_VALUE)	The value for a given chemical property	**Rate constant** of a given chemical compound having the value **3X10^{-5}**
condition-value (COND_VALUE)	The value for a given condition when the chemical reaction occurs.	**pH** being measured to be **5**.
value-unit (VALUE_UNIT)	The unit of a given value.	**5 mg/L**
property-compound (PROP_COMP)	The property associated with a given chemical compound.	**Rate constant** of a reaction associated with the compound **Chlortoluron**

For the purpose of conducting the text annotation, a Python package and web application known as Prodigy [4] was used. In Prodigy, the labels are defined and the dataset containing the raw samples is provided. One can label entities as well as the dependencies between entities in the tool's user interface. Therefore, the datasets containing the raw samples were inputted to the tool and the user interface was utilized to conducted the text annotations. An example of a text annotation using Prodigy is provided in Figure 3. The annotated samples were then verified by a researcher with a strong domain expertise to ensure that the annotations were accurate and did not include any human errors.

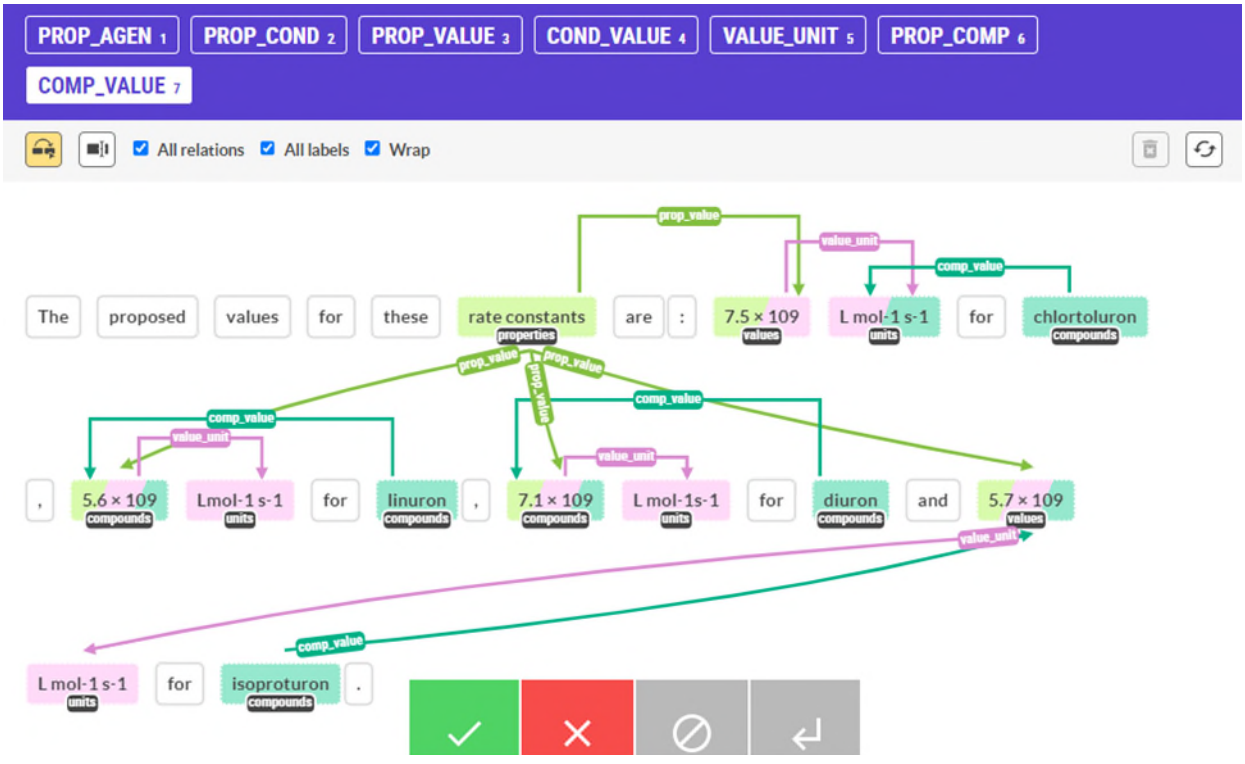


Figure 3: Text annotation example using Prodigy. Entities assigned to labels, provided in Table 1. Dependencies between entities assigned to labels provided in Table 2.

The application of this data preparation step led to the generation of 186 annotated text samples. The total number of samples for each entity label can be found in Table 3.

Table 3: Total number of entity samples per label, as available in 186 annotated texts

Label	Total Samples
agents	160
compounds	120
conditions	74
properties	320
units	98
values	130

3.2 Language Models

Typically within NLP, there are various processing operations performed on the annotated texts prior to its input into the neural networks for training or making a prediction. These include:

- *Tokenization*: Parsing or separating text (can be words, numbers or punctuation marks, into tokens.
- *Lemmatization*: Converting the token into its basic form, as you would encounter in a dictionary. For example, if a token is “flying”, the *lemma* form would be “fly”.
- *Part-of-speech* tagging: Assigning the part-of-speech of a given word, such as nouns, verbs etc.
- *Named Entity Recognition (NER):* As discussed before, these are entities within the text that can be named. The operation would involve the detection of these entities are assigning them to labels as trained to do so.
- *Parser*: This is related to the dependencies between entities. The parser operation would involve the detection of the dependencies based on the examples learned and assigning these linkages to provided labels.

For many of these processing operations, these are typically conducted specifically for a given language, such as English and Dutch, many of these operations have been conducted and stored, and do not required to be repeated. This is relevant for tokenization, lemmatization and part-of-speech operations. As a result, NLP models that include these operations can be reused to customize the remaining operations, and identify a specialized NLP application that is desired.

In this study, the Python package named SpaCy [5] was used. SpaCy provides the different operations as part of processing pipelines within a NLP model. These pipelines are already trained on large datasets associated with the language that the model is catered to. A user can then utilize the trained pipelines to either fine-tune components of it by provided additional annotated data, or directly utilize the models to make predictions. In our investigation, only the NER pipeline was fine-tuned. The parser (dependencies between entities) was not trained specifically for our case. This was due to the small dataset utilized for this study. For a language model to learn the dependency between entities can be a complex and time intensive process which requires numerous annotated samples to ensure

acceptable accuracy (often more than 500). Therefore, the model training was restricted to NER to evaluate whether using such specific datasets containing scientific texts on chemical compounds and properties can provide promising results. The remaining pipelines associated for tokenization, lemmatization and tagging was used as provided by SpaCy. The models and pipelines for the English language are trained on different corpus sizes, that is, the amount of text data that has been preprocessed and used to train a model to learn the syntaxes and structure of the natural language.

In this study, a SpaCy model trained on a medium-sized corpus of the English language was used. This was seen as sufficient as the use case of identifying specific information from scientific texts on chemical compounds is very specialized. The corpus size can only support the model in understanding the semantics associated with the English language, with respect to the structure and grammar. The main source of learning for the NER pipeline is associated with the annotated texts provided. Furthermore, the size of the corpus significantly influences the computational time for training as well. The fine-tuning of the NER pipeline was utilized the annotated data acquired from the data processing steps detailed in Section 3.1.2. The dataset was shuffled and then split into a training and a validation set with a 80/20 ratio, where the higher portion of the annotated samples was used for training. The model was validated using the smaller test set. Furthermore, the model performance was evaluated using three specific metrics – Precision, Recall and F1 Score, which have been defined below:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Precision is a value between 0 and 1, where 1 indicates perfect precision. This metric provides information on the number of correctly predicted labelled entities compared to the total number of predictions made for that given label.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Recall is a value between 0 and 1, where 1 indicates perfect recall. Recall is a ratio of the correctly predicted positive label entities to the total number of labelled entities in the observed dataset.

$$F1\ Score = 2\ X\ \frac{Precision\ X\ Recall}{Precision + Recall}$$

F1 score is the harmonic mean of the precision and recall, with a score of 0 and 1 being the lowest and highest, respectively. The F1 score provides a grouped metric by providing a balance between both precision and recall. Such a score can be significant in cases of imbalanced datasets, which can be a cause of concern considering some labels have lesser examples compared to others. This can be seen in Table 3, where the number of samples of 'conditions' is significantly lower than other entity labels such as 'properties'.

3.3 Trained NER Model Performance

In Table 1, the performance metrics of the best NER model on the test set after training on the annotated samples is provided. The ground truth used for assessment is the labelling performed by a human expert during the text annotations. Interestingly, the model performance is highly accurate, irrespective of the imbalance of number of examples that were prevalent for each entity label, as can be seen in Table 34.

Table 4: Precision, Recall and F1 Scores of a trained NER model on the test set for each entity label

Label	Precision	Recall	F1 Score
Agents	0.95	0.94	0.94
Compounds	0.91	0.89	0.89
Conditions	1.0	1.0	1.0
Properties	0.91	0.92	0.91
Units	0.96	0.96	0.96
Values	0.95	0.95	0.95

The model was able to classify with high accuracy all entities. An example of the classifications done by the model on the test set is illustrated in Figure 4. The results obtained are promising and indicate the strength of such language models to be tailored to user-specific needs when provided specialized datasets, such as scientific text related to chemical compounds and their reactive properties. The method proposed has been shown to be capable of extracting information from scientific texts related to chemical compounds and the discussed reaction conditions and numeric results. This allows the user to access semi-structured information in an automated manner and can also lead to a much faster process, as scientific information can be inputted to the model and receive the desired information. The promising NER results also provides strong base that can be used to fine-tune the parser pipeline. But it should be noted that the result may differ when applying the method to other applications, due the complexity of texts in these texts. The learning of the dependencies between entities is strongly associated with accurate identification and labelling of entities. A good performing NER pipeline suggest that training a parser model to also learn dependencies between entities that are specific to chemical compounds and properties. might be a promising activity.

Figure 4: Example of NER model classifications of the different entity labels on the test set. The individual colours represent the different entity labels – blue=properties, green=agents, yellow=compounds, red=conditions, purple=units and grey=values.

3.4 Concluding Remarks

In this analysis, scattered information that was retrieved from abstracts of scientific journal papers via a SCOPUS search was prepared and used to train the NER pipeline of a language model. The information was first annotated by manually labelling elements in the text to different user-defined labels that represent various features of the chemical experiments conducted in the scientific literature. This annotated and prepared data was then used to train language models that were evaluated to assess its performance of classifying the different entity labels. It was observed that the language model with the fine-tuned NER pipeline conducted highly accurate classifications of the various labels. However, it must be noted that the overall process is time-consuming. The annotation of text samples is labor-intensive and requires careful consideration of the context of the samples. This can lead to errors and requires constant verification by personnel with domain expertise. The training of models has been made relatively simple with the various open-source tools available but still requires a specialized skillset to be able to tailor such language models to the need of the users. Furthermore, to achieve a specialized NER pipeline that provided good performance is time consuming and is still not fit for deployment and directly utilized by end-users. The information provided is still semi-structured and the training of a parser pipeline is still necessary to achieve further advancement in this method. During the course of the project, the world saw explosive progress and availability of commercially available large language models (LLMs) such as ChatGPT and Bard that are extensively trained on millions of samples from the internet. It was foreseen that the direct usage of such models for our specialized use-case warrants more immediate attention prior to extending the specialized NLP application that is being developed from the ground up. Furthermore, such commercially available models are also available to be fine-tuned to specific use cases. As a result, it was concluded to first investigate the use of LLMs which has been detailed in Section 4.

4 Information extraction from scientific papers using large language models

4.1 Motivation

While the approaches developed and tested in Chapter 3 have been successful to some extent, we cannot overlook their limitations. Namely, manually annotating, analyzing and validating a large amount of texts is time-consuming and can be error-prone, while the approaches can also be inflexible and difficult to maintain.

During the implementation phase of this project (2023 Q1), a new NLP breakthrough, the so-called large language model (LLM), emerged as a promising solution for (scientific) literature information extraction. LLM is a type of deep learning that uses pre-trained language models as the basis for training new models for specific tasks, including named entity recognition and relation extraction. Readers can refer to the OpenAI's cookbook [6] or the BTO project – Large Language Models (to be finished by 2024 Q1) for more information on LLMs.

One of the main advantages of LLMs is that they require less raw data and resources for re-training or applying the model for particular cases. This is because the pre-trained language models have already learned a great deal about languages, particularly the syntactic and lexical features, and can potentially be fine-tuned for specific tasks with relatively little additional data and effort. Additionally, LLMs can be easily updated and adapted to new domains, making them more flexible and easier to maintain than custom-built models such as those described in Chapter 3.

Given this, we customized and applied the model of GPT 3.5 Turbo to conduct the same task as in Chapter 3, i.e., extracting information of chemical compounds from approximately 8,000 paper abstracts.

4.2 The Application of Large Language Model to Scientific Papers

4.2.1 Introduction of GPT 3.5 Turbo

Though the training data of the GPT 3.5 Turbo model only includes information up to September 2021, we do not aim to find real-time information using GPT models. Instead, we provide texts (paper abstracts) and use the GPT 3.5 Turbo model to find the needed information from the given texts. The GPT 3.5 Turbo model can take up to 4,000 tokens (c.a., 3,000 words) and provides an efficient API connection, which fits our case study for processing abstracts (often with fewer than 500 words). Meanwhile, the GPT 3.5 Turbo model cannot process images (neither can the model described in Chapter 3), which implies that texts are the main information to be extracted while using it in literature summarization.

4.2.2 Workflow

We designed a workflow consisting of four key steps, which include:

1. Defining the role of GPT-based virtual assistant
   The first step in our workflow is to define the role of the GPT-based virtual assistant. We do this by specifying its role either as a physicist, a chemist, or a biologist (Fig 5). By defining its role, we can better direct its performance towards providing useful information on chemical compounds.

```
# -----
system_role = [
    {"role": "system", "content": "You are a Physicist, Chemist, Biologist."},
]
```

Figure 5. Definition of GPT's role

2. Educating the virtual assistant to understand key concepts
   Once we have defined the role of the virtual assistant, we educate the virtual assistant to understand key concepts , including the reaction rate constant, the second-order reaction rate constant, quantum yield, etc. (see one example in Fig 6). This step is crucial because it ensures that the virtual assistant is familiar with the terminologies and concepts that we often use in a given domain, thereby avoiding any misunderstandings and inaccuracies in the information it finds.

```
system_edu_ones = [
    # https://en.wikipedia.org/wiki/Reaction_rate_constant
    {"role": "user",
     "content": "What is reaction rate constant?"},
    {"role": "assistant",
     "content": "In chemical kinetics,"
                " a reaction rate constant or reaction rate coefficient (k)"
                " quantifies"
                " the rate and direction of a chemical reaction."},
```

Figure 6. Educate the virtual assistant with one example of the reaction rate constant.

3. Sending appropriate prompts to the virtual assistant for information searching
   Then, we can directly ask questions from the virtual assistant about the information to be searched. We add a request of 'do not generate fake data' at the end of the prompt to ensure that we only receive accurate information.

```
'08.2': f"From the following text,"
        f" First, extract all coefficient value and corresponding coefficient name, reaction compound,
        f" Second, summarize in markdown table format with four columns;"
        f" Third, if no numerical coefficient value is found, then write NAN;"
        f" Do not generate fake data.",
```

Figure 7. Example of trial prompts.

4. Post-process information
   After receiving the information that we need, it still needs to undergo post-processing to ensure that information is accurate and reliable. This involves double-checking whether the information can be found in the original text and removing it if not. We also need to correct any inappropriate text formats that are commonly derived from the original text. For example, we came across 109 M-1s-1 in the text, which was corrected to 10^9 $M^{-1}s^{-1}$ for both the magnitude and the units by post-processing.

It should be noted that this workflow is not only applicable to chemistry-related literature but also to any other domains where one needs to find and extract information from a large corpus of text, making the process more efficient and reliable. Other applications require the user to customize prompts in step 3 by (optionally) educating GPT, specifying information-related questions and to customize post-processing in step 4 according to corresponding information. In general, by defining the role of the virtual assistant, educating it on key concepts, sending prompts for information searching, and post-processing information, one can ensure that the information extraction can be automated.

4.2.3 Results and discussion

The workflow described in the previous section was applied to 5,000 abstracts. The results were manually filtered by a domain expert for three types of (photo)chemical reaction rate constants that are used in advanced oxidation processes for water treatment: molar extinction at 254 nm (ext_254, the amount of photons that are absorbed by a chemical compound), quantum yield at 254 nm (QY_254, the fraction of absorbed photons that result in a chemical reaction of that chemical compound) and reaction rate constant of a compound with ozone (kO3). Also, only results

of single organic compounds were selected. For a few abstracts, results from the original paper were manually added. Only a small fraction (~1-2%) of the abstracts contained useful numerical information. The results were compared to an existing database of KWR containing the historically collected rate constant values of chemical compounds, The compiled information was programmatically converted to a table with columns indicating the name of compounds, agents, conditions, values, and units.

	C	D	E	F	G	H	I	J
1	Name	Compound	Agent	Condition	Value_str	Value_num	Value_fnd	Units
26	Quantum Yield	Estrone	UV radiation (253.7 nm)	Stationary	0.065 mol	0.065	TRUE	mol/Einstein
125	Quantum Yield	Estrone (E1)	UV radiation (253.7 nm	Continuous		0.107	TRUE	dimensionless
126	Quantum Yield	17 beta-estradiol	UV radiation (253.7 nm	Continuous		0.035	TRUE	dimensionless
127	Quantum Yield	estriol	UV radiation (253.7 nm	Continuous		0.029	TRUE	dimensionless
128	Quantum Yield	17 alpha-Ethinylestradiol	UV radiation (253.7 nm	Continuous		0.034	TRUE	dimensionless
129	Quantum Yield	17 beta-estradiol	UV radiation (253.7 nm	Stationary		0.016	TRUE	dimensionless
130	Quantum Yield	estriol	UV radiation (253.7 nm	Stationary		0.015	TRUE	dimensionless
131	Quantum Yield	17 alpha-Ethinylestradiol	UV radiation (253.7 nm	Stationary		0.018	TRUE	dimensionless

Fig 8. A snapshot of structured database for the quantum yield generated from the GPT model.

1. Accuracy

The accuracy of searching results is evaluated by comparing the values from KWR's database and those found by the GPT model from the literature. Figures 9, 10, and 11 show comparisons of three compounds. Note that the literature presents widely varying values for individual reactions. We calculated the mean values and ranges for each compound. The mean values, as shown in three figures for molar extinction (ext_254), quantum yield (QY_254) and reaction rate constant with ozone (kO3), are close in most of casesNevertheless, the difference of the mean values and the ranges (i.e., the error bars) can also be observed, which is presumably due to different conditions under which the reactions take place. The mean results appear reliable and meets our expectations.

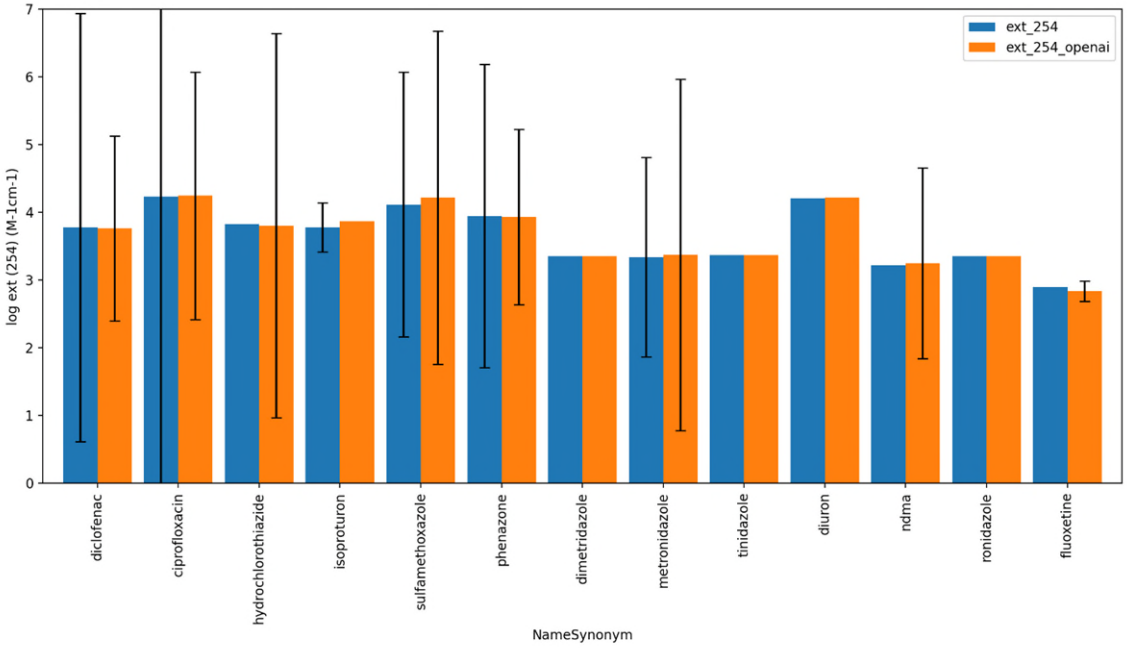


Figure 9. Verification of the results for log ext(254) comparing KWR's database values (blue) to GPT-compiled values (orange).

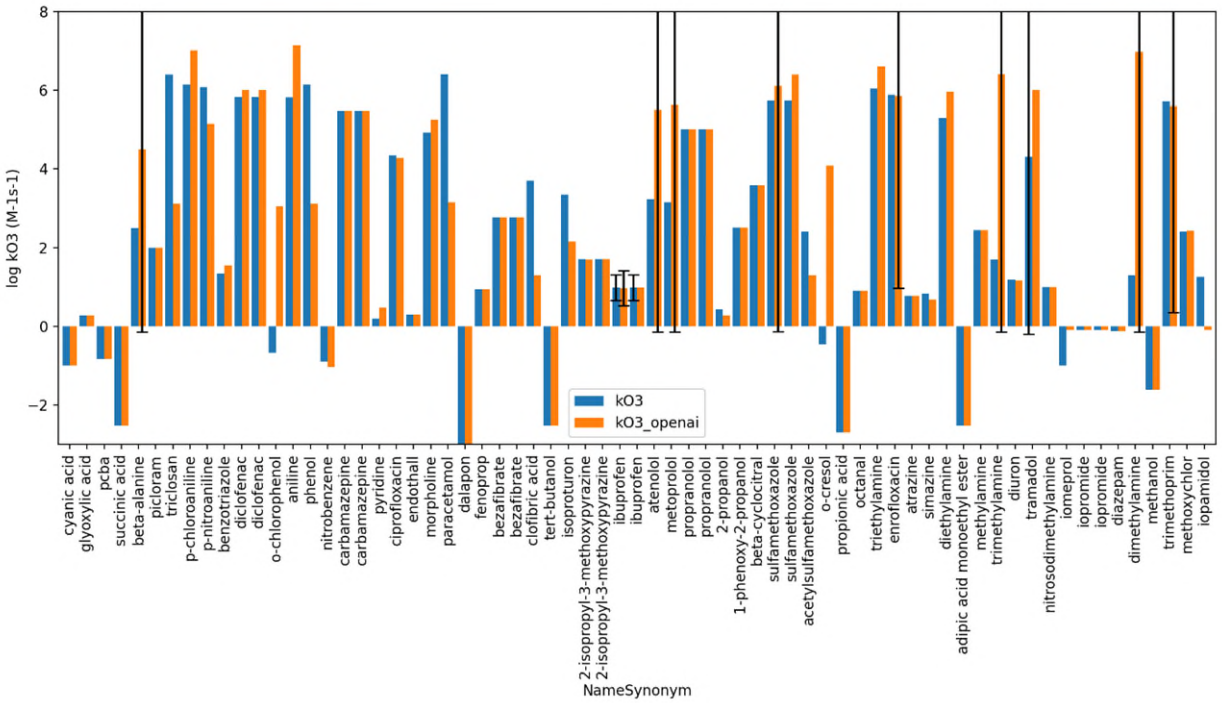


Figure 10. Verification of the results for kO_3, comparing KWR's database values (blue) to GPT-compiled values (orange).

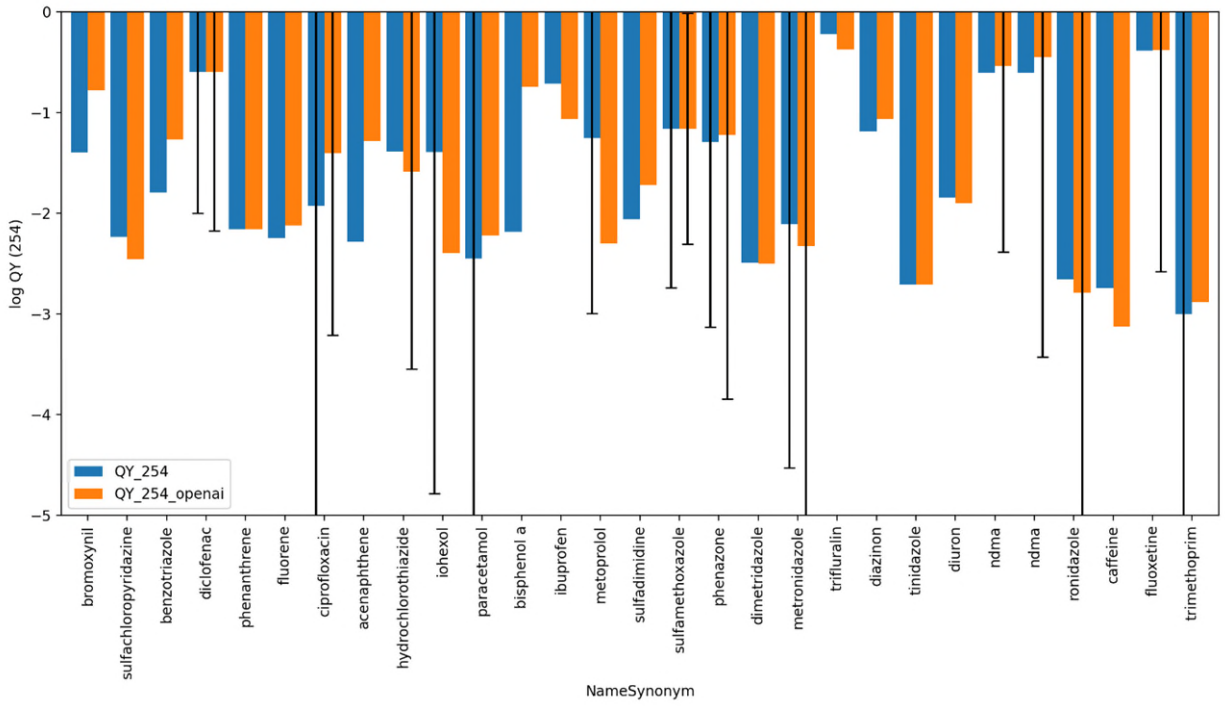


Figure 11. Verification of the results for QY_254, comparing KWR's database values (blue) to GPT-compiled values (orange).

2. Coverage

   In general, the use of the GPT model can substantially expand the required database for studying chemical compounds, which implies that the manual searching for rate constants values can be replaced by AI to a large extent. According to the study, any information that is well documented in unstructured and plain texts can be found and converted into structured texts.

3. Limitation

   It should be noted that GPT is still unable to retrieve and process information from non-plain material (e.g., tables and images) using their API at this stage. The functionality, however, is available from ChatGPT, where we just need to upload a PDF file and request GPT4 to extract information in tables. We anticipate that this functionality will be available soon for the API of all GPT models.

4. Use of results

   Using the results generated here, the database with chemical reaction rate constants (quantum yield, molar extinction and ozone reaction rate constants) was extended. The amount of unique compounds was increased by ~25% for QY_254 and ext_254, and more than doubled for kO_3 (see BTO 2024.011). The extended database was used to train new QSPR models that are used in water treatment process models (see BTO 2024.011).

5. Cost and time

   To apply such a workflow to any other applications, the following cost and time should be considered:

- The cost of using OpenAI' API

The use of the GPT 3.5 turbo model cost 0.003 cents per 1,000 tokens (i.e., 750 words) for inputs and 0.004 cents per 1,000 tokens for outputs. For example, we screened 8,000 paper abstracts in this project, which cost around 12 US dollars. However, it is worth noting that the cost may vary, depending on the size of contexts and the complexity of the task.

- Time

It took approximately 20 hours with frequent communication of the OpenAI's API to process all 8,000 abstracts. This means that around 400 abstracts can be processed within an hour, assuming a stable internet connection. However, the time required may vary depending on the complexity of the task, i.e., the number of prompts, the length of the abstract, or the density of needed information.

6. Prepare a script with appropriate prompts

In order to effectively use OpenAI's API, it is necessary to prepare appropriate prompts and (Python) scripts. Trial prompts may also be necessary to test whether the desired information can be obtained. Additionally, a script is needed to call OpenAI's API and format the output. It roughly takes one to two weeks to prepare such a script with a junior to mid-level Python programmer based on the experience in this project.

4.3 Concluding remarks

Compared to conventional language models, the use of LLMs has so far proven to be the most effective way to extract information from texts. The method developed in this case study can also be applied to other domains where a large amount of information needs to be searched.

5 Conclusions and recommendations

5.1 Conclusions

This study first investigated the automation of literature downloading and processing of scientific publications. This allows the collection of much larger quantities of numerical information from the literature than previously feasible. Following that, we proposed two methods to extract specific information on chemical reactions, namely, the self-developed NER model and the GPT model. The self-developed model still requires a large number of annotations done by hand by a domain expert while the GPT model, pre-trained on an extremely large corpus, can find information more effectively and efficiently. From our study, we recommend the use of such a GPT-based workflow for processing scientific publications.

5.2 Implications and recommendations

The workflow developed in this project (see Section 4.2.2) is a generic one, which is not limited to searching chemical information only. In fact, this workflow also fits in other cases by adjusting key words for searching, modifying questions for information extraction, and formatting and validating results. We strongly recommend researchers and practitioners consider this usage in literature searching and processing at the beginning of other research projects, with required time and cost (see Section 4.2.4). For instance, practitioners from water utilities and water labs can use the proposed method and workflow to widely search for information of PFAS, microplastics, etc., and automatically summarize the information in a tabular database for the following analysis.

5.3 Future directions

In this study, we only considered paper abstracts, which mostly contain plain texts only. It should be noted that much more useful information is also in full texts, including tables and figures. This will be part of future research of us to include full texts in the text mining. Meanwhile, we also aim to expand the applicability of this method to more research fields, including but not limited to hydrology, chemical and biological water quality, etc.

Following this research, it also shows that we can further utilize LLMs to help us with efficient literature searching (often at the beginning of a project), fact searching and knowledge management (during and after a project), and accelerate research progress and acquire new projects.

6 References

1. https://dev.elsevier.com/
2. https://guides.lib.berkeley.edu/information-studies/apis
3. https://pypi.org/project/pyscopus/
4. https://prodi.gy/
5. https://spacy.io/
6. https://cookbook.openai.com/
7. https://platform.openai.com/docs/models
8. B.A. Wols, R.P.J. Hoondert, P.S. Bauerlein (2024), Zeer Zorgwekkende Stoffen (deel 1) – clustering, bemonstering en toxiciteit, BTO 2024.010, KWR Water Research Institute, Nieuwegein, The Netherlands.
9. B.A. Wols, W. Siegers, D. Vries (2024), Zeer Zorgwekkende Stoffen (deel 2) – zuivering, BTO 2024.011, KWR Water Research Institute, Nieuwegein, The Netherlands.
10. V. Post, B. van der Grift, M. van der Schans (2024), Verslag van veldmetingen en historische meetgegevens over afbraak van organische microverontreinigingen in grondwater, BTO 2024.013, KWR Water Research Institute, Nieuwegein, The Netherlands.