Automated Extraction of Domain-Specific Information from Scientific Publications

Philipp Kief*, Clarissa Marquardt†, Katja Nau†, Steffen Scholz†, and Andreas Schmidt*†
* Department of Computer Science and Business Information Systems,
Karlsruhe University of Applied Sciences
Karlsruhe, Germany
Email: philipp.kief@gmx.de, andreas.schmidt@hs-karlsruhe.de
† Institute for Automation and Applied Informatics
Karlsruhe Institute of Technology
Karlsruhe, Germany
Email: clarissa.marquardt@kit.edu, nau@kit.edu, scholz@kit.edu, schmidt@kit.edu

***Abstract*—As the number of scientific publications in many subject areas continues to increase, it is becoming more and more important to support researchers in filtering out relevant information from papers and to identify relevant papers as well. In the present work, the field of nanotoxicology is used to investigate how dictionary-based disambiguation and extraction of entities of the domain can be implemented and how information on entire scientific papers can be extracted. By developing an analysis tool, it can be shown that the automated analysis of scientific publications in the field of nanotoxicology can be realized in basic terms. The analysis tool is based on the General Architecture for Text Engineering (GATE), D3.js and additional Node.js services, as well as Angular.js and represents an application that can be controlled intuitively by the scientists and provides a suitable user interface to visualize the extracted information in an aggregated way.**

***Keywords*–Named entities; Domain specific entities; Entity cooccurrence; Visualization.**

I. Introduction

The number of potentially relevant publications in the field of nanotoxicology is increasing at a rate that is difficult to manage even for experts in the respective fields. As a consequence, more time is needed to extract certain information from different sources. Here, mostly local document collections or public platforms like PubMed or ResearchGate can be mentioned. This trend will certainly continue due to the distribution possibilities on the Internet and in particular due to the great research potential at least in the field of nanotoxicology. Therefore, approaches that automatically attempt to extract semantic information from scientific work are becoming increasingly important for researchers to keep track of other research in their field.

A common difficulty is that ambiguities (example “Paris”: city, ship, band, hotel, biological term for a plant genus, etc.) must be resolved. This process is called Named Entity Disambiguation (NED). Entities are words or phrases that represent a real object (e.g., persons, organizations, places, etc.). These objects do not necessarily have to exist physically, but can also be abstract, such as a year or date. These entities are commonly referred to as *Named Entities*, where here the proper name of an entity is meant (e.g., the name of a person, where the person is an entity) [1]. In this paper, we will investigate exemplarily how an automatic extraction of the entities from the field of nanotoxicology can be carried out. Therefor, it is important to recognize the relevant entities in the respective domain. In the case of nanotoxicology these are, for example, chemical substances, diseases and medical terms.

In the context of this work, a concept is to be developed, with which entities relevant for the nanotoxic domain can be extracted from text. For this purpose, existing knowledge databases such as eNanoMapper [2], Medical Subject Headings (MeSH) [3], etc. will be used. Based on these results, various analyses, such as cooccurrences, analysis of temporal trends, clustering of similar documents, etc., are to be carried out by means of suitable visualization.

A. Information Extraction

Information Extraction (IE) is a sub-area of Natural Language Processing (NLP) and is used to extract information from a large amount of unstructured data. After extraction, the resulting data is presented in structured form. Based on this structured data it is possible to conclude further knowledge afterwards. IE is mainly used in areas where a very large amount of text, from which information needs to be extracted in a very short time, are available. An example for IE is the extraction of entities from over 500 different news feeds [4]. In this case the relationship between different entities are to be determined (e.g., how often appear the entities “Donald Trump” and “Angela Merkel” together in the collection of news articles).

The first phase of the IE process contains domain-independent tasks. These include *sentence splitting*, *tokenization*, *morphological analysis* and *Part-Of-Speech* (POS) tagging [5], which are only dependent on the language but not on the domain. Since a sentence represents a semantically closed unit and the words within a sentence have a strong relation to each other, it is necessary to perform a sentence splitting as one of the first steps within IE. In addition, tokenization transforms all words and punctuation marks into so-called tokens. A token represents the smallest text element of a text and is the basis for further IE steps. In a further morphological analysis, the properties of a token are determined. Properties are the POS and the lemma of a word. Lemma is the basic form of a word under which it can also be found in a dictionary. The POS is the type of word (e.g., noun, adjective, verb, preposition, etc.). In the subsequent second phase, domain-specific components such as NED or the identification of related entities are located.

B. Named Entity Recognition

Named Entity Recognition (NER) can be seen as a subcategory of IE. NER aims to identify the entities in a text that are relevant to a subject area. In biomedical text, the names of persons, addresses, telephone numbers as well as symptoms, diseases, medications or anatomical features are of importance [6]. Nevertheless, it depends on the concrete application of an analysis tool which entities have to be extracted from a text corpus. The extraction of the entities is therefore important, since they contain a large part of the information about the content of a document.

C. State of the Art

There exist already many different tools and frameworks that can extract information from documents. For example, the National Library of Medicine (NLM) has developed a program called MetaMap [7] to extract biomedical sections of a text to match them to the concepts of the UMLS (Unified Medical Language System). UMLS is a terminology that comprises over 2 million names on 900,000 biomedical concepts and is freely available for research purposes [8]. Another tool is PolySearch2 [9], which is a web-based tool that can extract the relations between biomedical entities. It is mainly designed to formulate queries according to the scheme "Given X, find all associated Ys". An example query can look like this: "Find all diseases associated with Bisphenol A". The results of the queries are based on data from MEDLINE [10], PubMed [11] and 14 other biological databases such as UniProt [12], Drug-Bank [13] and Human Metabolome Database [14].

D. Contribution of the paper

The main contributions of this paper are the following: (1) Extraction of all relevant information from large document collections in the domain of nanotoxicology. (2) Further processing of the extracted data such as the recognition of relationships between entities or temporal dependencies as well as the calculation of the relevance of an entity to a document. (3) Implementation of a generic analysis tool into which a corpus of documents can be loaded and analyzed. Results of the analysis are made available by means of visually descriptive visualizations. Although this is shown by the example of nanotoxicology, the procedure can be regarded as a blueprint for any domain, since only the steps for the extraction of the entities have to be adapted.

II. Conception

This section shows how a document corpus can be read and analyzed, so that the collected information can later be displayed graphically via suitable visualizations.

A. Preprocessing

Preprocessing is the part of the application that is responsible for extracting all the necessary data from the documents and storing it in a structured data format. The extracted data must be available in such a form that it forms a suitable basis for later calculations and analyses. This process is executed once for each document and the data will be stored in a database.

The first step is the importing of the documents into the tool. Since the documents are available in different formats such as PDF, Word documents or images, they must be converted into a uniform, processable format. The aim is to have the text of the documents as raw text, i.e., without markup elements or other syntax that is responsible for the layout. Many documents contain additional information in the form of metadata, which is also stored in the document. Metadata includes, e.g., the name of the author, the creation date, the date of the last modification, various keywords, the title or a short description of the content. For example, keywords provide very precise information about the content of a document and the creation date can be used to classify the document by year.

Finally, the potential retrieval of documents from Open Access sources should also be mentioned. Some of them have potentially useful Application Programming Interfaces (APIs), so that publications on a particular topic can be automatically retrieved via the analysis tool and then analyzed. The results of this analysis can be sent to the user by e-mail. Such a scenario can save the user a lot of time, as the tool can search for potential publications and trigger the analysis fully automatically without further manual steps.

B. Extraction of Named Entities

Once a document has been loaded and the content has been converted to the appropriate format, the next step is to extract the relevant entities. In order to recognize entities, a comparison with different dictionaries is necessary. The dictionaries typically contain a number of terms per entity. For example, for the geographical area, there are dictionaries that list the names of the cities of the world. If a word or phrase from the text is found in such a dictionary, it can be marked as a city-entity. By the information which entity a word is, further information can be obtained in the following steps. Important entities in the context of this work are names of persons, places, organizations, dates, addresses as well as domain-specific technical terms of the nanotoxicology domain. A separate dictionary was compiled for the latter by collecting 855,237 entries from different databases such as MeSH [3], eNanoMapper [2], Nanoparticle Ontology [15], Springer Nature (SN) SciGraph and the DaNa Glossary.

If entities are recognized in the text, further information about this entity is also stored. As a result, the positions at which the entity occurs in a document are recorded. A position contains the information about the document, the index of the surrounding sentence and the position of the word within the sentence. Since an entity usually occurs more than once in a document, a list of item information is assigned to this entity.

C. Text Normalization

Different methods can be used to find terms with similar meaning. Based on the lemmatization algorithm [16], two words, which differ from each other only for grammatical reasons, can be adapted. In lemmatization, a word is reset to its basic form, which is also called Lemma [16]. Thus, for the words "studies" and "studying" the lemma "study" can be determined. In this case, both words can be summarized under the term "study". An alternative to lemmatization is the stemming algorithm. A stemmer or stemming is generally understood as the truncation of the suffix of a word [17]. With the same word base and different suffix, many words usually have the same meaning (e.g., "connect", "connected", "connecting" or "connection") [17]. Both methods are similar and both can be used to summarize terms. Since the words are more consistently reduced to one form by stemming than by lemmatization, this paper will focus on stemming. As only a comparison between the words is made here, the shortened words do not necessarily have to be readable, as is the case with lemmatization. In the medical field, many abbreviations are used in scientific documents. In this case, the abbreviations should also be combined with the spellings written out as they can be referred to the same entity.

D. Relations between Entities

The situation that two entities occur close to each other is known as cooccurrence. The concept of this work is based on the assumption, that entities that often appear together (coocurrence) have a potentially strong relationship. In order to be able to measure this relationship, we must quantify the dependency, based on the cooccurences of two or more entities. Beside the frequency of this coccurence, also the distance between the entities, which form the cooccurence, must be considered. The smaller the distance between two entities, the stronger is their cooccurrence.

The concept envisions that as a preprocessing step for all existing entities located in a defined proximity, the respective distance to all other entities is calculated. A threshold-value must be defined for the maximum distance between two entities. The larger the value, the longer the calculations take, since the distance between more entities must be calculated. The maximum distance thus represents a compromise between the duration of the calculation and the coverage of all relationships. However, since widely spread entities have a minimal relationship to each other, a calculation does not have to be performed for all occurrences. For this work the limit value was set to 20 words, because the strong relationships between entities could be determined and the calculation speed is still acceptable high.

To measure the strength of the distance between two entities (E_1 and E_2), we developed a formula, that includes several factors. If the two entities are in the same sentence, only the distance between the words is calculated. The distance is given by subtracting the index of E_1 from E_2. If the entities can be found in different sentences (S_1 and S_2), the distance of the sentences to each other is additionally calculated here, i.e., how many sentences lie between entity E_1 and entity E_2. Finally, the distance of entities contained in different paragraphs (P_1 and P_2) will also be added to the formula. Each distance can be weighted differently with an exponent (here α, β and γ). The exponents are weighted in descending order because the distance between the entities has a strong influence on the strength, the distances between the sentences and the paragraphs should have a weaker influence. Finally, an inverse value is calculated from the sum of the differences, so that large distances lead to a lower strength and vice versa.

$$strength = \frac{1}{(E_2 - E_1)^\alpha + (S_2 - S_1)^\beta + (P_2 - P_1)^\gamma}$$

E. Identification of Relevant Documents

The relevance of a document to an entity should not only be based on the number of occurrences of the entity in a document, but should be calculated from a combination of several factors. If only the number of occurrences of an entity were taken into account when calculating relevance, this could result in the name or year of an author appearing very frequently in the bibliography at the end of a document and not reflecting the entire content of a document. From a frequently appearing name of a person in the bibliography, one can only conclude the fact that the author of the document has often used that person's literature. This does not mean that the document has anything to do with this person. Similar to Information Retrieval (IR) procedures, various factors should be taken into account, such as how often an entity appears in the text, whether it is part of the document title, where it appears in the document, how widely it spreads throughout the document, and whether it is listed in the keywords of a document. If an entity is part of the title or the keywords, the relevance for this document increases considerably. Title and keywords usually describe a document very concisely and contain essential entities to describe the content.

In addition, it is also conceivable to include the entities that are part of an abstract more strongly in the calculation of the relevance of a document. An abstract usually describes the content of a document in a few concise sentences. An entity extracted from it can thus have an important meaning for the content of the according document.

Another possibility that can be considered for the relevance of a document is the temporal frequency of an entity. Thus, for example, it can be determined in which years there was a certain trend in the use of a term. Documents, which in this case are no longer in the period of the trend, should be weighted less strongly, because they, for example, only report retrospectively on the trend and do not supply any more new knowledge.

F. Frequency History

There are always periods in which certain entities are used particularly frequently by authors in literature. Time trends can be determined from the frequency, with which an entity appears over several years in different documents. Researchers at Harvard University, for example, found that the term "slavery" was very widely used in the early 1860s during the civil war and the civil rights movement from 1955 to 1968, by analyzing over 5 million books from the 16th to the 20th century [18].

The investigation of the frequency of entities over a certain period of time allows the identification of trends and a temporal classification. When searching for specific entities, the user can find out in which years many documents deal with the entity or when the entity first appeared. In the periods when entities occur more frequently than average or rarely, further analyses can be carried out by the researchers in order to search for the cause.

G. Architecture

The architecture for an automated document analysis tool essentially consists of three components. Before the preprocessing can take place, the documents must be uploaded. Afterwards, the entities with their position information are extracted from the text and stored in a database. The processing service in turn retrieves the data from the database and sends it to the client to display it visually. The client can use a search to decide for which entity he wants to receive additional information and for which entity relevant documents should be displayed. An autocompletion helps the user find the right entities and the search can also be limited by search criteria such as entity type or number of related entities. A rough overview about the architecture is given in Figure 1.

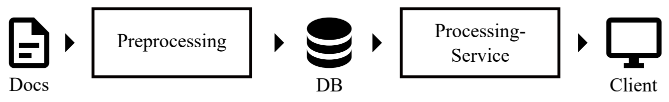


Figure 1. Components of the tool

The preprocessing step can be started automatically after the documents have been successfully uploaded into the tool. Depending on the number of documents, it may take some time to process the individual steps. During processing, the client should be shown a status, which document is currently being preprocessed and how long it will take. Since preprocessing consists of several smaller sub-processes such as tokenization, sentence separation, POS, comparison with dictionaries, calculation of relations, determination of the relevance of documents, extraction of metadata, recognition of abbreviations, stemming as well as the assignment of terms to an entity, these must be brought together to form a pipeline. Within the pipeline, the subprocesses are executed one after the other, since some of them also depend on previous ones.

The only component that is actually dependent on a specific domain is the comparison with the dictionaries. In order to be able to recognize technical terms in documents, it is necessary to create certain terminologies that can be used for comparison. Such an approach is, on the other hand, very generic, since for another domain only the dictionary has to be exchanged in order to identify the relevant entities from the documents. Thus, the analysis tool could also be used very well in other fields than nanotoxicology.

III. Implementation

The implementation of a prototypical analysis tool will be used to demonstrate how the analysis of scientific documents in the field of nanotoxicology can be realized. The application can be controlled centrally via the client, which is a web application based on the web framework Angular. A web application was chosen, because the client only has to make requests to the implemented services and displays the data on the user interface at the end. A web application does not need to be natively installed on a computer and is lightweight and fast. All preprocessing steps and calculations are handled by the backend. The backend in turn consists of a Node.js server, which answers the client's requests and initiates the necessary analysis steps of the documents. Since the stored information about the extracted entities from the documents partly have different attributes, the NoSQL database MongoDB was used. This database makes it easy to store data records as JSON objects, regardless of which keys are contained in the JSON object. The dataformat used is JSON because JSON is a very lightweight format for storing data and can be easily processed with Node.js. Also JSON is the native dataformat in MongoDB and very well suited for transmission via REST interfaces, which have also been implemented here.

A. Apache Tika

The toolkit Apache Tika [19] was used for the conversion of the PDFs, because it can not only extract the text but also recognize the metadata of a PDF file. A big advantage of this toolkit is the output in HTML format via the option "–html". In HTML output, paragraphs are syntactically marked with a <p> tag, so that in a next step in the pipeline, the paragraphs can be saved as entity information as well. Occasionally, words whose syllables were separated with a hyphen at the end of the line by a line break could no longer be correctly combined with Tika. Consequently, a regular expression is used to search for exactly these occurrences and the line breaks are removed by a script.

B. GATE - General Architecture for Text Engineering

A large part of the further steps of the preprocessing is realized by a Java-based tool called GATE [20]. It offers a variety of plugins that can be combined to a pipeline. A plugin can be individually added to a pipeline via a plugin manager. Since many plugins are already included by default, only a few plugins have to be added manually for the prototype. By default it includes tools for tokenization, gazetteers, sentence splitting or POS tagging. Each plugin can create annotations in the text and add additional information to them. An annotation represents a marking of a certain place in the text and enhances this with additional information. These can be very diverse and depend on the plugin that can add, change and remove new information in the form of properties (so-called "features").

Which plugins are used for the implementation of the tool can be seen in Table I. The Document Normalizer plugin normalizes various special characters in the texts, then the ANNIE English Tokenizer splits the words into tokens and the sentences are annotated by the RegEx Sentence Splitter. The ANNIE Gazetteer compares the tokens with basic dictionaries such as city names, addresses, people names, etc. and annotates recognized words as corresponding entities. The ANNIE POS Tagger applies POS-Tagging to all tokens and with the help of the ANNIE NE Transducer several so-called JAPE (Java Annotation Pattern Engine) rules are applied to the annotations. These rules ensure that some properties of the annotations are renamed (e.g., "minorType" to "gender") for better semantics. With the Transfer Original Markups plugin mainly the paragraph tags from the HTML output of Tika are converted into paragraph annotations. The Morphological analyzer plugin calculates the word root for each word and adds this information to the annotation. To recognize the annotations an Abbreviation Extraction plugin was developed, which implements the algorithm of S. Schwarz and M. Hearst [21] which is mainly designed for the recognition of abbreviations in biomedical text. This plugin also detects how an abbreviation is written out and adds this information to the according annotation. Abbreviations also need to be stemmed by an Abbreviation Stemmer plugin so that they can be more easily combined later. The last plugins in the pipeline are all those that are domain-specific for nanotoxicology, which is why they are marked with the prefix "NANO". These plugins perform the same tasks as the previous plugins only that they are directly designed for annotations that can be assigned to nanotoxicology. This is because, for example, specially designed dictionaries are used, which only contain technical terms from the domain or special JAPE rules, which have to adapt these annotations in order to obtain a better data structure later.

As a result of the execution of the GATE pipeline, a separate JSON document exists for each document in which all annotations are contained as objects. Based on several Node.js scripts, these JSON documents are combined into a single data set. The annotation objects are compared with each other on

TABLE I. PLUGINS OF THE GATE PIPELINE

#	Plugin	Functionality
1	Document Normalizer	Replace special characters
2	ANNIE English Tokenizer	Split words into tokens
3	RegEx Sentence Splitter	Annotate sentences
4	ANNIE Gazetteer	Detect entities from GATE
5	ANNIE POS Tagger	Recognize POS of word
6	ANNIE NE Transducer	Customizing annotations
7	Transfer original markups	Annotate HTML tags
8	Morphological analyzer	Recognize root word
9	Abbreviation Extraction	Extract abbreviations
10	Abbreviation Stemmer	Stemming of abbreviations
11	NANO Gazetteer	Extracting NANO entities
12	NANO Abbreviation Gazetteer	Extracting NANO abbreviations
13	NANO Transducer	Adapting NANO annotations
14	NANO Stemmer	Stemming NANO entities

the basis of their stemmed value and merged as appropriate. Based on these data sets, the relations between the annotations are calculated as already described in the conceptual part of this paper. As a result, for each annotation it is stored to which other annotations a very strong relation exists. The same is done for the calculation of relevant documents as well as for the calculation of the temporal classification of an entity. The calculated data is stored in a MongoDB in the form of JSON data records.

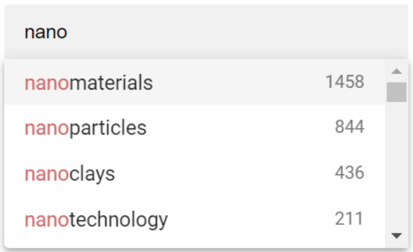
C. Orchestration of the processing tasks

To be able to start GATE directly by a request comming from the Node.js server, the Java API (GATE Embedded) is called. Thereby a configuration file is specified at the start of GATE, in which all plugins of the pipeline are defined. Since Tika and various Node.js scripts are executed in addition to GATE, a higher-level pipeline of process calls must be defined. This is realized via a Makefile. In the Makefile, the individual processes are configured as targets whose order is clearly defined when the Makefile is called.

D. Web application

The web application is based on the Angular.js web framework. A file upload was realized, with which the PDF documents can be uploaded. The PDF documents must always be part of a document collection. A document collection is created in the database after the upload. Furthermore, it is possible to edit the dictionaries via the web application or to create new dictionaries with new technical terms. This enables researchers to keep the identification of relevant terms up to date in the future. After a document collection has been created, the analysis can be triggered for it. The Web application triggers this with a request to the processing service. During the processing of the individual steps of the original processing, a status dialog is displayed in the Web application that shows which step is currently being prepared. This was realized by transferring the standard output of the process pipeline in the console to the Web application via a Web socket connection. The web application then interprets the logs in such a way that a process bar is generated out of it.

When the analysis of the document collection is finished, the user has the possibility to search the data set from the database for certain terms. The user is supported by an autocompletion function (see Figure 2). The number behind a search suggestion indicates how often this term occurs in the document collection.


nano

nanomaterials	1458
nanoparticles	844
nanoclays	436
nanotechnology	211

Figure 2. Autocompletion

After the user has clicked on one of the search suggestions, a new page will opened, displaying all available information about the selected entity (see Figure 3). The relationships of the selected entity to other entities are visualized in the form of a graph. The strength of the lines corresponds to the strength of the relationship between the entities. The nodes in the dark blue color represent direct relationships, while the nodes in the light blue color represent the most relevant entities of the dark blue nodes. This attempts to contextualize the entities within the graph. If required, additional information about the selected entity can be displayed in a panel next to the graph. This includes, for example, information about existing abbreviations as well as a descending list of the most relevant documents for this entity. The documents can also be opened or downloaded directly from this view.

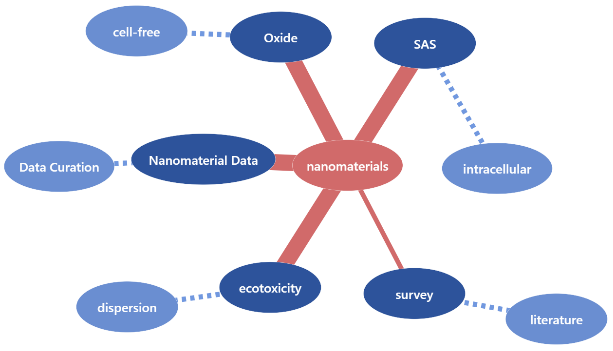


Figure 3. Visualization of entity relationships

Another visualization is to compare the most common entities of two document collections. A graph visualization is also generated, which displays the entities from both document collections in two different colors. In a third color, which represents a mix color tone of the other two colors, the intersection is displayed (see Figure 4). The intersection includes those entities that occur very frequently in both document collections. From this it can be concluded that the two document collections have a certain thematic intersection concerning these entities.

IV. EVALUATION

The prototypical implementation of a tool for the automated analysis of scientific documents showed that common concepts of information extraction can be combined to extract information from unstructured texts and to transform them

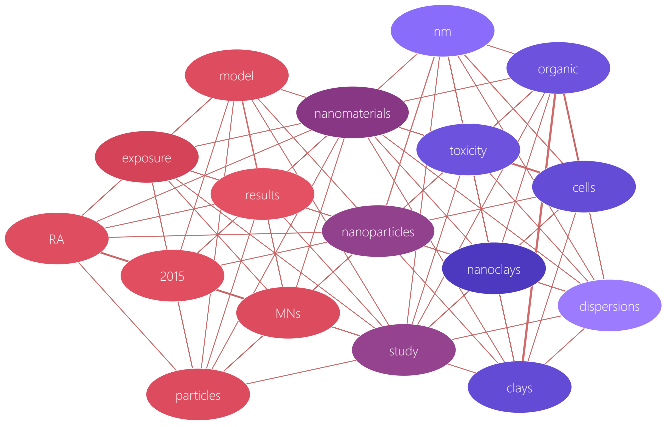


Figure 4. Comparison of document collections

into a structured, machine-processable data model. However, this concept is very much based on the quality of the dictionaries. If, for example, the terminology used in a particular department is not as good as in nanotoxicology, the quality of the results will also suffer. A good selection of dictionaries and terminologies is therefore crucial. The advantage of the implementation is the concept of the web application and the decoupled services in the backend, so that there are no hurdles during the installation of the software at the end user. The loose coupling of services and process steps in the pipeline makes it easy to add new components or replace existing ones in the future.

V. Conclusion and Further Work

The paper reported on an automated analysis of scientific publications on nanotoxicology. It was shown, with which concepts a solution can be implemented, which can facilitate the daily work of scientists when searching through large document collections. Several tools and technologies were presented, with which a prototypical software was developed. In addition, it was shown which possibilities are available to visualize large amounts of collected information in a suitable way.

As a further step towards automated document analysis, it can be explored whether the presented solution can also connect to online databases to automatically download and analyze new documents and information. The concept could even go to the point where the researcher is informed by e-mail that a new document is available after the analysis has been carried out. Certain filter rules could also define that the researcher is only notified if the document is relevant to him and his current research. In this way, further manual steps in literature research could be automated, making it easier for researchers to find relevant publications for their research.

Acknowledgements

The work presented in this paper was partly funded under the DaNa2.0 project funded by the German Federal Ministry of Education and Research (FKZ 03X0131).

References

[1] A. Siu, “Knowledge-driven entity recognition and disambiguation in biomedical text,” Ph.D. dissertation, 2017.

[2] J. Hastings, N. Jeliazkova, G. Owen, G. Tsiliki, C. R. Munteanu, C. Steinbeck et al., “eNanoMapper: harnessing ontologies to enable data integration for nanomaterial risk assessment,” J Biomed Semantics, vol. 6, 2015, p. 10, ISSN: 2041-1480.

[3] “Medical Subject Headings - The NLM’s curated medical vocabulary resource,” 2019, URL: https://www.nlm.nih.gov/mesh/meshhome.html [retrieved: 2019-07-21].

[4] A. Schmidt and S. Scholz, “Quantitative considerations about the semantic relationship of entities in a document corpus,” in Proceedings of the 51st Hawaii International Conference on System Sciences, Hilton Waikoloa Village, Hawaii, January 2-6, 2018, 2018, pp. 933–942, URL: http://hdl.handle.net/10125/50003 [retrieved: 2019-07-21].

[5] M. Rodrigues, “Advanced applications of natural language processing for performing information extraction,” Cham, 2015, URL: https://doi.org/10.1007/978-3-319-15563-0 [retrieved: 2019-07-21].

[6] H. Dalianis, “Clinical text mining : Secondary use of electronic patient records,” Cham, 2018, URL: https://doi.org/10.1007/978-3-319-78503-5 [retrieved: 2019-07-21].

[7] A. R. Aronson, “Effective mapping of biomedical text to the umls metathesaurus: the metamap program.” in Proceedings of the AMIA Symposium. American Medical Informatics Association, 2001, p. 17.

[8] O. Bodenreider, “The unified medical language system (umls): integrating biomedical terminology,” Nucleic Acids Research, vol. 32, 2004, URL: http://dx.doi.org/10.1093/nar/gkh061 [retrieved: 2019-07-21].

[9] Y. Liu, Y. Liang, and D. Wishart, “Polysearch2: a significantly improved text-mining system for discovering associations between human diseases, genes, drugs, metabolites, toxins and more,” Nucleic Acids Research, vol. 43, 2015, URL: http://dx.doi.org/10.1093/nar/gkv383 [retrieved: 2019-07-21].

[10] “MEDLINE,” 2019, URL: https://www.nlm.nih.gov/bsd/medline.html [retrieved: 2019-07-21].

[11] “PubMed,” 2019, URL: https://www.ncbi.nlm.nih.gov/pubmed/ [retrieved: 2019-07-21].

[12] “UniProt,” 2019, URL: https://www.uniprot.org/ [retrieved: 2019-07-21].

[13] “DrugBank,” 2019, URL: https://www.drugbank.ca/ [retrieved: 2019-07-21].

[14] “HMDB,” 2019, URL: http://www.hmdb.ca/ [retrieved: 2019-07-21].

[15] D. G. Thomas, R. V. Pappu, and N. A. Baker, “Nanoparticle ontology for cancer nanotechnology research,” Journal of Biomedical Informatics, vol. 44, no. 1, 2011, pp. 59–74, ontologies for Clinical and Translational Research, URL: http://www.sciencedirect.com/science/article/pii/S1532046410000341 [retrieved: 2019-07-21].

[16] C. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval. Cambridge University Press, 2008, URL: http://www.informationretrieval.org [retrieved: 2019-07-21].

[17] M.F.Porter, “An algorithm for suffix stripping,” 1980, URL: https://tartarus.org/martin/PorterStemmer/def.txt [retrieved: 2019-07-21].

[18] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett et al., “Quantitative analysis of culture using millions of digitized books,” Science, vol. 331, no. 6014, 2011, pp. 176–182, URL: http://science.sciencemag.org/content/331/6014/176 [retrieved: 2019-07-21].

[19] C. A. Mattmann and J. L. Zitting, Tika in Action. Manning, 2011.

[20] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, M. Dimitrov, M. Dowman et al., Developing Language Processing Components with GATE Version 8. University of Sheffield Department of Computer Science., 2018, URL: https://gate.ac.uk/sale/tao/tao.pdf [retrieved: 2019-07-21].

[21] A. S. Schwartz and M. Hearst, “A simple algorithm for identifying abbreviation definitions in biomedical text,” University of California, Berkeley, 2003, URL: http://biotext.berkeley.edu/papers/psb03.pdf [retrieved: 2019-07-21].