PLOS ONE

RESEARCH ARTICLE

PubMedPortable: A Framework for Supporting the Development of Text Mining Applications

Kersten Döring[1], Björn A. Grüning[2], Kiran K. Telukunta[2], Philippe Thomas[3], Stefan Günther[1]*

1 Pharmaceutical Bioinformatics, Institute of Pharmaceutical Sciences, Albert-Ludwigs University, 79104 Freiburg, Germany, **2** Bioinformatics, Institute of Computer Science, Albert-Ludwigs University, 79110 Freiburg, Germany, **3** Language Technology Lab, German Research Center for Artificial Intelligence, DFKI GmbH, 10559 Berlin, Germany

* stefan.guenther@pharmazie.uni-freiburg.de

Citation: Döring K, Grüning BA, Telukunta KK, Thomas P, Günther S (2016) PubMedPortable: A Framework for Supporting the Development of Text Mining Applications. PLoS ONE 11(10): e0163794. doi:10.1371/journal.pone.0163794

Editor: Robert Guralnick, University of Colorado, UNITED STATES

Received: February 22, 2016

Accepted: September 14, 2016

Published: October 5, 2016

Data Availability Statement: All relevant data are within the paper and its Supporting Information files. Additionally, the framework is also available at the public repository GitHub (https://github.com/KerstenDoering/PubMedPortable/).

Funding: SG received funding for this project by the German National Research Foundation (DFG, Lis45). The article processing charge was funded by the German Research Foundation (DFG) and the University of Freiburg in the funding programme Open Access Publishing.

Abstract

Information extraction from biomedical literature is continuously growing in scope and importance. Many tools exist that perform named entity recognition, e.g. of proteins, chemical compounds, and diseases. Furthermore, several approaches deal with the extraction of relations between identified entities. The BioCreative community supports these developments with yearly open challenges, which led to a standardised XML text annotation format called BioC. PubMed provides access to the largest open biomedical literature repository, but there is no unified way of connecting its data to natural language processing tools. Therefore, an appropriate data environment is needed as a basis to combine different software solutions and to develop customised text mining applications. PubMedPortable builds a relational database and a full text index on PubMed citations. It can be applied either to the complete PubMed data set or an arbitrary subset of downloaded PubMed XML files. The software provides the infrastructure to combine stand-alone applications by exporting different data formats, e.g. BioC. The presented workflows show how to use PubMedPortable to retrieve, store, and analyse a disease-specific data set. The provided use cases are well documented in the PubMedPortable wiki. The open-source software library is small, easy to use, and scalable to the user's system requirements. It is freely available for Linux on the web at https://github.com/KerstenDoering/PubMedPortable and for other operating systems as a virtual container. The approach was tested extensively and applied successfully in several projects.

Introduction

Large progress has been made in the field of text mining and natural language processing (NLP) in the biomedical domain [1]. This includes identification of protein-protein interactions [2], drug-drug interactions [3], compound-protein interactions [4, 5], and their

Competing Interests: The authors have declared that no competing interests exist.

connection to diseases [6]. Nevertheless, research efforts are still hindered by a lack of standardised ways to process the vast amount of data. This matter can be divided into two issues. First, there is the problem of interoperability between NLP components for named entity recognition (NER) and relation extraction methods. Second, literature-related data needs to be accessible for large-scale applications.

In this publication, the problem of data accessibility is approached with a combination of relational database and full text index. While a full text index can be built for complex Boolean text queries, a relational database is suitable for storing all meta information of PubMed articles and collecting statistics.

The issue of connecting different software solutions for natural language processing is covered by the implementation of an interface to the BioC interchange format. The aim of PubMedPortable is to enable users to develop text mining applications and use cases with very basic programming knowledge. This is understood in terms of a stand-alone application without dependency on web services, but with the possibility to query them if desired. Any tool in a customised workflow supporting BioC input and output modules can be applied to perform NLP tasks independently, but there are also other directly usable data formats, as shown in the remainder of this article.

Related work on Software Interoperability

Several proposals have been made to tackle this research area, but only a few of them have obtained wide interest in the community, namely the Unstructured Information Management Architecture (UIMA) [7, 8], the General Architecture for Text Engineering (GATE) [9], and the BioC XML data format [10]. UIMA defines Text Analysis Engines (TAEs), the text processing software modules, and a common analysis structure (CAS), the XML-based input and output format for TAEs [10]. U-Compare is a Java Web Start application that offers drag-and-drop construction of workflows for UIMA-compatible NLP tools [11]. The GATE Developer is an integrated development environment in Java similar to U-Compare [7]. GATE provides an interface to UIMA as well. A large-scale example for the application of UIMA is the AusTalk corpus [12], a 3,000 hour auditory-visual corpus of Australian English. Its data sets as well as well as other corpora are accessible via the Alveo Virtual Laboratory [13], a web platform consisting of tools in different programming languages, ensuring interoperability with UIMA. As an alternative to the complex architectures of UIMA and GATE, docrep [14] represents a lightweight document representation framework for serialisation of textual data with linguistic annotations.

BioC uses a minimalistic approach, as only the data format of XML files is defined in a document type definition (DTD) file, accompanied by the user-specific semantics of data and annotations, which are described in an extra key file. The interoperability is ensured within the BioC workflow, defining an Input Connector to read and an Output Connector to write BioC XML data. The interface to these BioC classes is implemented in several programming languages [10, 15].

Rak *et al.* claimed that there is a tendency towards workflow construction platforms, but that their software dependencies on a source platform can restrict the development process [7]. Therefore, web services became popular to solve NLP tasks, especially because of the Representational State Transfer (REST) architecture. This was their motivation to develop Argo, an UIMA-based online text mining workbench, supporting different data formats like CAS and BioC. Argo offers a range of web service components to be used in a workflow, without additional programming efforts [7]. There are advantages and disadvantages for choosing UIMA or BioC, but the integration of UIMA-compatible modules can be considered as a more

complex process than generating a BioC-compatiple XML document [15]. Therefore, PubMedPortable implements a BioC interface. End users can easily apply and combine BioC NLP tools to arbitrary PubMed data sets, whereas developers can focus on sophisticated pipelines, including e.g. machine learning approaches.

Approaches to Process PubMed Data

At the beginning of 2016, PubMed consisted of more than 25 million records and the number increases quickly. Considering the issue of how to deal with this large amount of data and to apply NLP methods effectively, quite a few published as well as unpublished approaches exist.

There are efforts to simplify literature searches in PubMed and to support text annotation. Two of the most recent web service developments are OntoGene [6] and PubTator [16], which provide a BioC interface.

Working with PubMed publications on a local machine is possible, too. The complete data set of PubMed can be downloaded as XML files from the NLM FTP server, including example data sets. The user can also apply the NCBI interface to download a set of PubMed XML files related to a specific search term [17]. Biopython can be used to connect to this interface, named EFetch. It also contains a library to parse PubMed XML files [18]. The content of XML files does not have to be processed multiple times if the text elements are stored in a database. The LingPipe project is implemented in Java and offers a library to parse PubMed XML files and build a full text index with Lucene [19]. It contains a short tutorial about loading the abstract and title texts into a MySQL relational database in a version from 2010. It is worth noting that the PubMed XML schema is updated annually. Biopython and LingPipe are considered sophisticated tools, but this also means that it is a complex task to modify existing and develop new functions to process and index all recently available PubMed XML attributes.

There are finished implementations building a relational database from PubMed XML files. According to the tendency towards web services as mentioned by Rak *et al.* [7], these approaches were already published around 10 years ago [20, 21]. Yoo *et al.* describe a complex system for downloading PubMed and PMC articles in XML format, storing them in a MySQL relational database, and searching the documents with a Lucene full text index [21]. Unfortunately, their service is not hosted anymore. In 2004, Oliver *et al.* compared different approaches to loading PubMed XML files into a relational database [20]. This included an Oracle 9i and an IBM DB2 relational database, combined with Java and PERL code. There is an unpublished update version from 2010 using Java 6 and MySQL 5.1 [22].

In PubMedPortable, this SQL schema is completely adapted and slightly modified, but combined with object-relational mapping (ORM) in Python to generate a PostgreSQL database from PubMed XML files. That means, changes of SQL tables or columns can be introduced directly in the parser itself, and the whole uploading process can be upscaled to the number of desired CPU cores. PubMedPortable implements a Xapian full text index similar to Yoo *et al.* [21], as any PostgreSQL column can be indexed with this technology with only a few lines of code. Currently, abstract titles and texts, MeSH terms, keywords, and chemical substances are indexed in the standard implementation. There is also a modified version only indexing abstract titles and texts. Using a Xapian index offers fast and straight forward keyword and context search, as shown by the use cases described in the results section.

While PubMedPortable was tested in Ubuntu and Fedora, there is also a one-click-solution based on Docker [23], so that the relational database and the full text index can be used in more operating systems. Docker is similar to a virtual machine and easy to deploy. Therefore, PubMedPortable helps to standardise the way of processing large literature data sets and making them accessible for text mining applications.

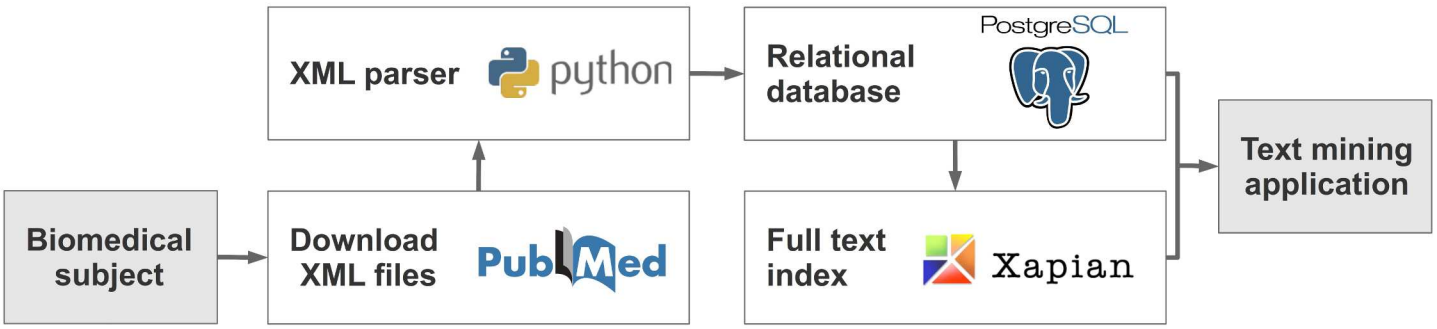


Fig 1. PubMedPortable workflow. 1) Download XML files from PubMed. 2) Parse and upload data into a PostgreSQL relational database. 3) Build a Xapian full text index. 4) Develop text mining applications.

doi:10.1371/journal.pone.0163794.g001

Methods

Data Accessibility

The basic requirements are an installation of Python, Xapian, and PostgreSQL, as well as around 300 GB free disk space in case of processing all PubMed XML files. Fig 1 shows the general workflow for loading PubMed XML files into a relational database and generating a full text index.

PubMedPortable is usable via a command-line interface and requires PubMed XML files as input. A database needs to be created and configured at first. The tables are built based on user-provided data. The SQL schema can be seen in the GitHub project folder. All PubMed XML attributes are transformed into PostgreSQL tables and columns with an ORM approach using a SAX parser. After processing the PubMed XML files, a full text index is built by querying titles, abstracts, MeSH terms, keywords, and chemical substances from the relational database. These installation steps can also be executed at once using the virtual container Docker without installing additional software packages.

Based on this data environment, phrase and Boolean searches can be executed with user-defined terms. It is possible to generate charts and statistics by combining the full text search with SQL queries. A range of examples and detailed installation instructions are precisely described on the GitHub project page.

BioC Interface

A multitude of BioC compatible tools can be applied by exporting articles from PubMedPortable to BioC format. This includes the NER tools from Wei *et al.*, identifying chemicals/species, diseases, mutations/variations, and genes/proteins [24], included in the PubTator web service [16]. Tokenisation, part-of-speech (POS) tagging, and sentence parsing can be e.g. performed with the BioC NLP pipeline from Comeau *et al.* [25].

The BioC workflow as shown in Fig 2 implies that any tool supporting this input and output format can add annotations to a document, supporting the idea of interoperability. The PubMedPortable documentation describes how MeSH terms from the relational database can be added to a BioC XML document.

Fig 3 shows the output of invoking PubTator and merging the results with MeSH term annotations as a BioC document. All implementation details can be found in the GitHub project folder.

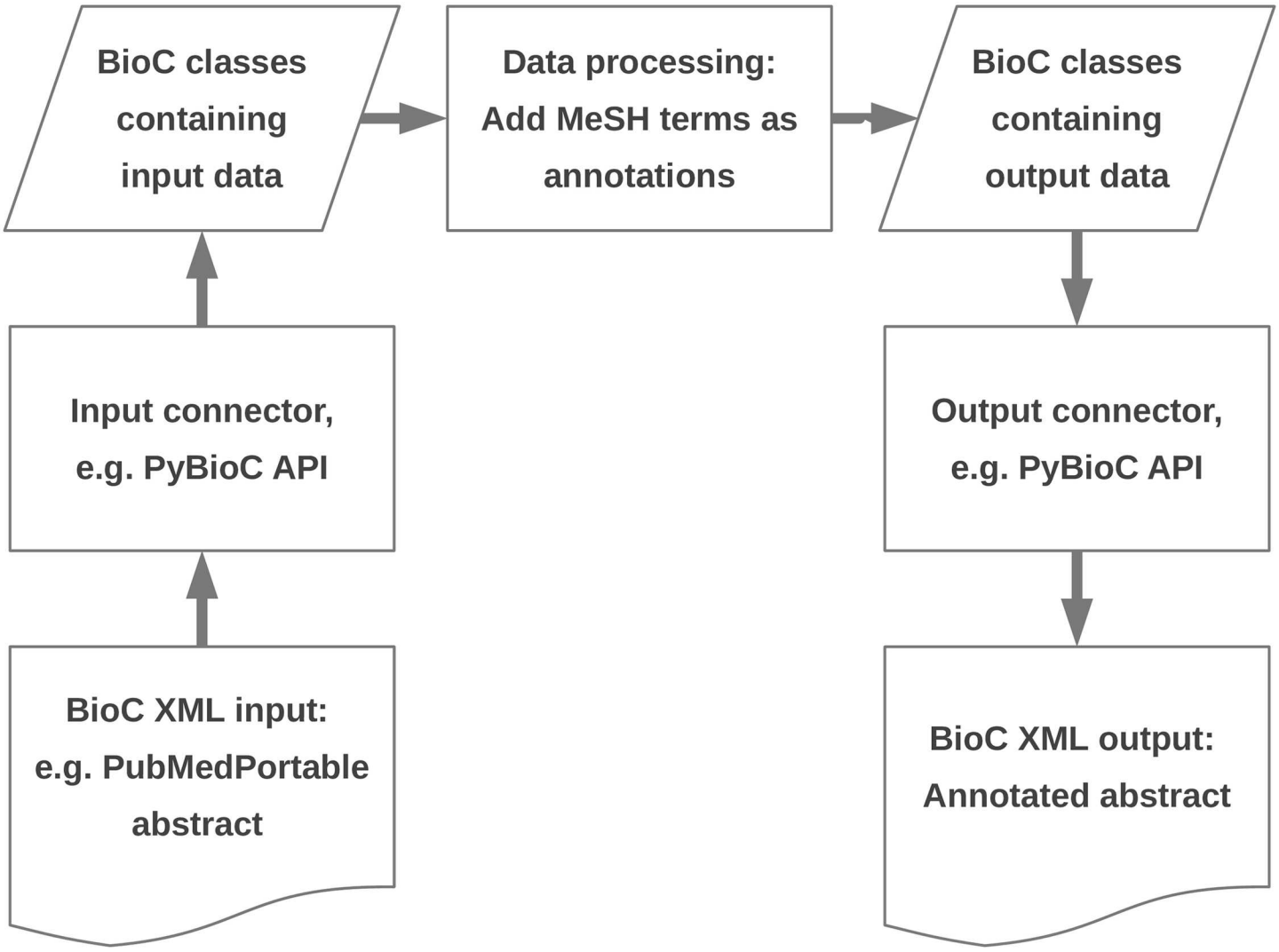
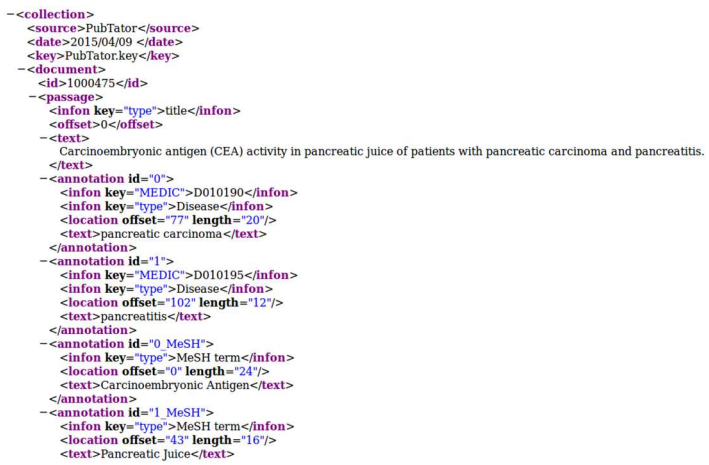


Fig 2. General BioC workflow. This is the minimalistic approach from Comeau *et al*. [10] with the example how to add MeSH terms to BioC PubMed titles and abstracts from the PubMedPortable PostgreSQL database.

doi:10.1371/journal.pone.0163794.g002

As mentioned in the introduction, the requirements for BioC are a DTD file to define the XML document structure and a key file to explain which structural elements are used in a particular BioC XML document. XML infon elements are used to differentiate between distinct XML elements. Every infon element contains a key and a value. This basic element can refer to the document sections title or text, to token IDs, or to NER annotations. Every BioC XML file starts with naming the original creation source and the creation date. Within the XML schema of Fig 3, the text collection consists of PubMed articles with their PubMed ID as document ID. Each document consists of a title and the abstract text, if given. In this case, the file PubTator. key describes the semantics used by this web service. For the sake of clarity, Comeau *et al*. intend that only one type of annotation should be used within one BioC XML file consisting of several documents [10], but they can also be combined as illustrated here.

Fig 4 shows a code snippet how to read in and iterate over BioC XML elements with their annotations.



```
−<collection>
   <source>PubTator</source>
   <date>2015/04/09 </date>
   <key>PubTator.key</key>
 −<document>
    <id>1000475</id>
  −<passage>
     <infon key="type">title</infon>
     <offset>0</offset>
   −<text>
      Carcinoembryonic antigen (CEA) activity in pancreatic juice of patients with pancreatic carcinoma and pancreatitis.
     </text>
   −<annotation id="0">
      <infon key="MEDIC">D010190</infon>
      <infon key="type">Disease</infon>
      <location offset="77" length="20"/>
      <text>pancreatic carcinoma</text>
     </annotation>
   −<annotation id="1">
      <infon key="MEDIC">D010195</infon>
      <infon key="type">Disease</infon>
      <location offset="102" length="12"/>
      <text>pancreatitis</text>
     </annotation>
   −<annotation id="0_MeSH">
      <infon key="type">MeSH term</infon>
      <location offset="0" length="24"/>
      <text>Carcinoembryonic Antigen</text>
    </annotation>
   −<annotation id="1_MeSH">
      <infon key="type">MeSH term</infon>
      <location offset="43" length="16"/>
      <text>Pancreatic Juice</text>
```

Fig 3. Excerpt of a BioC XML document. The document ID 100475 is a PubMed ID. PubTator annotations are shown with infon elements that contain the key *type* with the value *Disease* and the key *MEDIC* referring to a MeSH ID, such as *D010190* for the given disease *pancreatic carcinoma*. The PubMedPortable MeSH term annotations are shown with the annotation IDs *0_MeSH* and *1_MeSH* to make them distinguishable from the normally iterating PubTator annotation IDs. They were added after calling the PubTator web service.

doi:10.1371/journal.pone.0163794.g003

Results and Discussion

Performance

For the complete XML data set available in 2015 with a size of 114 GB, it took 10.5 days to build the PostgreSQL relational database and another 27 hours to generate the full text index using a 2.8 GHz quad-core processor. The time of the indexing process and the size of the index depends on the number of extracted fields. A modification of the PubMedPortable scripts, including only abstract titles and texts, but not MeSH terms, keywords, and substances, led to a runtime of 10 hours. The size of the full text index also decreased from 154 GB to 124 GB. It is difficult to compare the runtime to the results from Oliver *et al.* [20] due to different hardware and software system requirements, but increasing computational resources will speed up this process in general. Using 48 CPU cores with 2.1 GHz reduced the result calculation time of 10.5 days to 20 hours.

Pancreatic Cancer Data Set

The PubMedPortable documentation refers to a small data set of 23,258 PubMed IDs (272 MB) related to pancreatic cancer, which is processed in a few minutes. The documents were selected by performing a phrase search with the term pancreatic cancer using the NCBI web service. Pancreatic cancer is one of the most dangerous cancer types. Currently, the only way to cure a patient is surgery, beside several therapeutic strategies that cannot significantly increase survival rates [26]. The research progress in this area can be supported with text mining methods, e.g. by covering findings about gene-disease and compound-protein relationships.

Use Case: BioC Applications

To get a first impression of important genes or proteins, drugs, and diseases related to pancreatic cancer, these entities were automatically identified in the approximately 23,000 abstracts.

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-

# Read BioC XML elements with PyBioC

# PyBioC API
from bioc import BioCReader

# open BioC XML file and DTD file with XML structure definitions
bioc_reader = BioCReader("annotated_text_BioC.xml", dtd_valid_file="BioC.dtd")
# read files
bioc_reader.read()
# get documents from BioC XML file (PubMed abstracts)
docs = bioc_reader.collection.documents

# iterate over documents
for doc in docs:
        # show document ID (PubMed ID)
        print "PubMed ID:",doc.id
        # iterate over passages - PubMed titles and abstracts
        for passage in doc.passages:
                # show passage type
                print "Text type:", passage.infons['type']
                # iterate over annotations for each passage and show information
                for annotation in passage.annotations:
                        print "Annotation ID:", annotation.id
                        print "Annotation Type:", annotation.infons['type']
                        print "Annotation Text:", annotation.text
                        print "Offset and term length:", annotation.locations[0]
        # line break
        print "\n"
```

Fig 4. Read BioC elements. All BioC XML elements can be read with the BioC API. The script refers to the left part of the workflow shown in Fig 2. Iterating over the given annotations as shown in Fig 3 will e.g. show *Annotation ID: 0*, *Annotation Type: Disease*, *Annotation Text: pancreatic carcinoma*, and *Offset and term length: 77:20*.

doi:10.1371/journal.pone.0163794.g004

Gene and protein synonyms were extracted with GeneTUKit [27]. Disease terms were annotated with the DNorm [28]. Both tools were used without retraining their models on an annotated corpus. Chemical compounds were identified with tmChem [29] via the PubTator web service [16]. DNorm and tmChem as stand-alone applications work with BioC as well as with an individual tab-separated format. GeneTUKit takes texts in a pseudo XML format as input and generates tab-separated annotations as output. The software was embedded into a short pipeline to be used with PubMedPortable, described in a GitHub side project [30].

The number of abstracts for each entity was counted based on summarising the synonyms for each identifier provided, e.g. Entrez-GeneID numbers. One identifier probably refers to different synonyms, and some annotated examples did not receive such a code. From all extracted entities, the 150 most commonly used identifiers were extracted to be illustrated in a word cloud. Fig 5 shows the steps to create the word cloud shown on Fig 6. All details can be found in the project wiki on GitHub.

One approach to evaluate the main entities shown in Fig 6 is to compare diseases and gene or protein synonyms with information provided by the database OMIM [31]. Chemicals identified as drugs can be reviewed in DrugBank [32]. These databases are focused on human diseases, their interrelated gene mutations, and how to treat them. One assumption is that if e.g.

```
1. Gene and protein named-entity recognition
2. Disease named-entity recognition
3. Chemical named-entity recognition
4. File concatenation (CSV)
5. Count entities, summarized by their identifiers and represented by one synonym
6. Generate logarithmic values (first 150 entities)
7. Create word cloud
```

Fig 5. Documentation to generate a word cloud using PubMedPortable. Different tools and different data formats might be used for named-entity recognition. Tab-separated files (CSV) with PubMed ID, synonym, and identifier in each line are used to collect all abstracts in which a match for identifier-specific synonyms appeared.

doi:10.1371/journal.pone.0163794.g005

specific oncogenes like *KRAS* are presented in OMIM, the NER search should result in a reasonable number of publications. The relative frequencies of search terms are displayed in the word cloud in Fig 6 by different sizes. The absolute numbers of the 15 most frequently appearing diseases, genes or proteins, and chemicals are summarized in Table 1. The majority of terms is clearly related to cancer or inflammation processes. *KRAS* is part of the largest displayed genes. The protein *p16* refers to the gene *CDKN2A*. *PANC-1* and *MIA PaCa-2* refer to cell lines. Another point to mention about the word cloud is that Entrez-GeneID numbers are organism-specific. Therefore, the large *EGFR* example refers to the human gene (GeneID

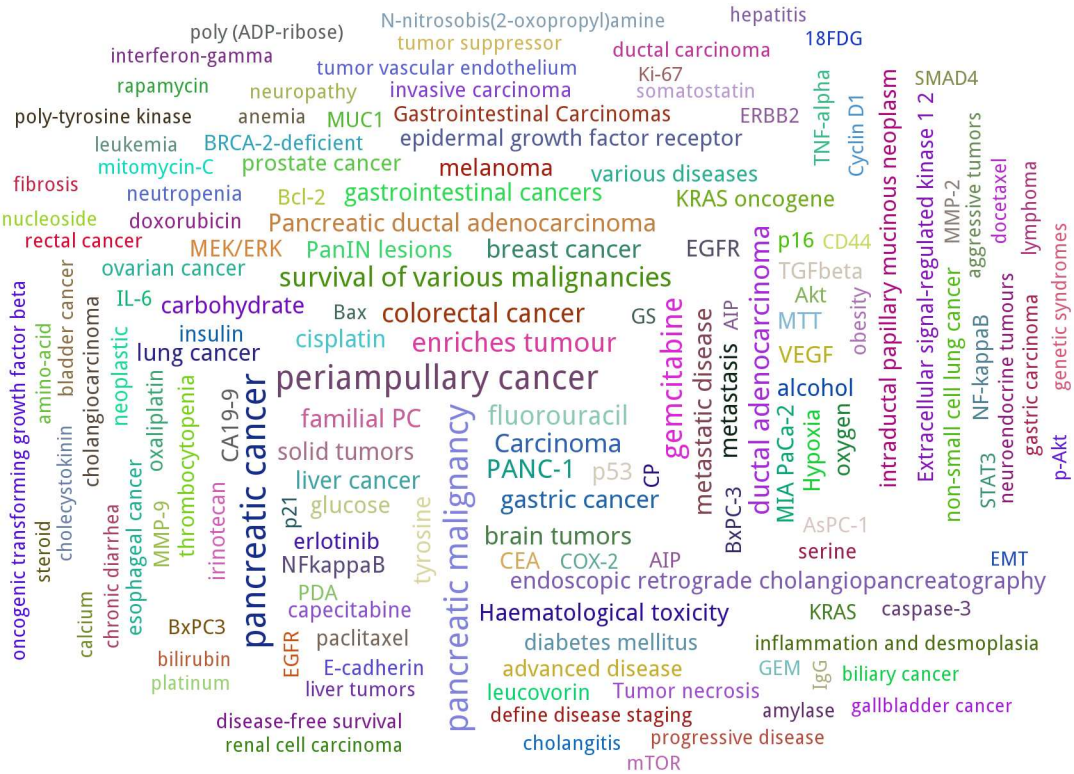


Fig 6. Genes, proteins, chemicals, and diseases related to pancreatic cancer. The 150 most frequently appearing entities in terms of their number of abstracts were identified with DNorm [28], GeneTUKit [27], and PubTator [16]. Fig 5 shows the steps to generate this word cloud.

doi:10.1371/journal.pone.0163794.g006

Table 1. 15 most commonly found entities from the 3 entity types disease, gene/protein, and chemical with the number of identified PubMed abstracts.

Rank	Disease	#	Gene/protein	#	Chemical	#
1	pancreatic cancer	22,823	*PANC-1*	884	gemcitabine	2,806
2	periampullary cancer	13,302	*p53*	409	fluorouracil	1,055
3	pancreatic malignancy	9,364	*VEGF*	380	cisplatin	442
4	ductal adenocarcinoma	1,544	*EGFR*	361	tyrosine	420
5	survival of various malignancies	1,279	*MIA PaCa-2*	307	carbohydrate	383
6	enriches tumour	1,260	*CA19-9*	294	alcohol	381
7	colorectal cancer	1,082	*NFkappaB*	277	glucose	373
8	Carcinoma	818	*CEA*	273	MTT	354
9	breast cancer	747	*MEK/ERK*	272	erlotinib	311
10	gastric cancer	677	*epidermal growth factor receptor*	266	oxygen	263
11	familial PC	647	*TGFbeta*	259	oxaliplatin	227
12	metastatic disease	602	*KRAS oncogene*	258	insulin	223
13	brain tumors	506	*NF-kappaB*	237	capecitabine	217
14	Pancreatic ductal adenocarcinoma	506	*p16*	229	leucovorin	214
15	liver cancer	444	*Bcl-2*	229	paclitaxel	208

doi:10.1371/journal.pone.0163794.t001

37455) and the slightly smaller term *epidermal growth factor receptor* to *Drosophila melanogaster* (GeneID 1956). However, the process of gene normalisation needs to be treated cautiously in the case of short texts as PubMed titles and abstracts. Gemcitabine (DrugBank ID DB00441) is the most frequently occurring chemical, a nucleoside analog commonly used in chemotherapy [26]. Periampullary cancer is located close to the ampulla of Vater (pancreatic duct), possibly related to jaundice [33]. Pancreatic ductal adenocarcinoma is the most frequently appearing type of pancreatic cancer and the most common lethal cancer [34]. Fig 6 and Table 1 illustrate that texts referring to pancreatic cancer are also related to colorectal, breast, gastric, and liver cancer, as well as diabetes, and other disease-associated terms. The importance or relevance of a relation between pancreatic cancer and a selected entity can be inferred by considering the number of co-occurrences or analysed with more sophisticated methods, e.g. the approach of kernel methods as applied for the identification of protein-protein interactions [2] and compound-protein interactions [35].

This use case illustrates how the BioC interoperability can be applied for fast development of prototypic text mining applications with PubMedPortable in terms of software modularity. More BioC-related software and text corpora can be found in the overview of the BioCreative IV interoperability track [36] and within the BioC track of the BioCreative V challenge [37]. Wei *et al*. provide an overview of NER applications published within the last ten years [24].

Use Case: Querying PubMedPortable Data Sets Related to Pancreatic Cancer

According to the OMIM review about pancreatic cancer mentioned in the last subsection, three substantially involved genes are *KRAS*, *CDKN2A*, and *BRCA2*. While Fig 6 shows their relative frequencies in comparison to other search terms, the timelines in Fig 7 illustrate the absolute number of publications per year. In contrast to Fig 6, different organism-specific Entrez GeneID numbers were summarized. Compared to the other two genes, the *KRAS* timeline grows strongest. There is actually no slope for the *CDKN2A* timeline, but the amounts are

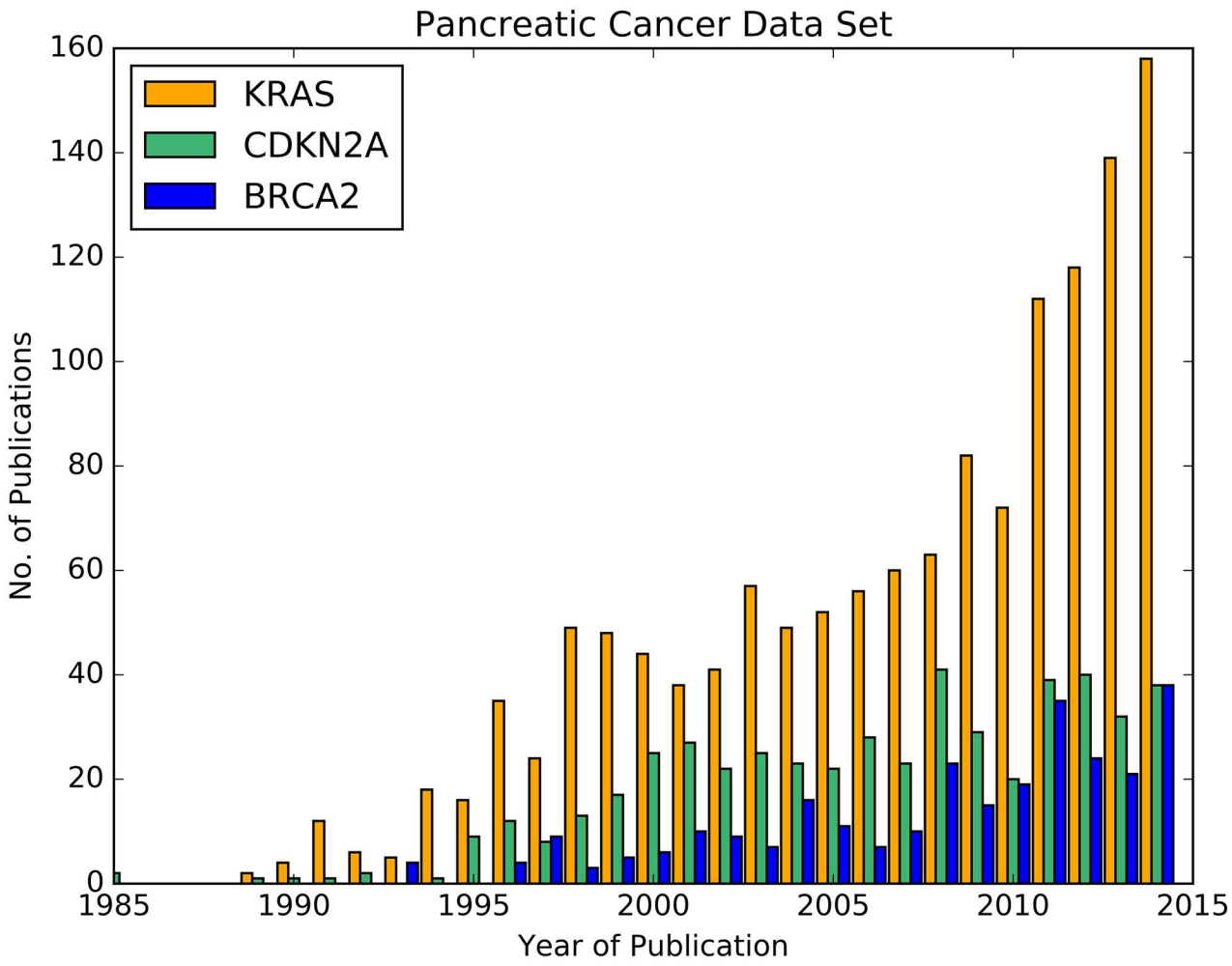


Fig 7. Timelines for the publications of the genes *KRAS*, *BRCA2*, and *CDKN2A* until 2014. The PubMed IDs for these three genes were extracted from the list of entities resulting from step 4 in Fig 5. The publication years were selected from the PubMedPortable database.

doi:10.1371/journal.pone.0163794.g007

consistently on a higher level than the ones of *BRCA2*, although converging in 2010. Both genes show a rather low number of publications compared to *KRAS*. One reason for this outcome is the role of *KRAS* in the regulation of cell proliferation and its higher specificity to pancreatic cancer than in case of *BRCA2* and *CDKN2A* [38]. In combination with the word cloud, these examples present ideas to visually inspect the number of publications of specific entities.

One way to investigate the context of selected search terms is to use the PubMedPortable function to generate an HTML page with highlighted entities. The full text index can be searched with keyword queries for this purpose. The PubMedPortable documentation describes examples, such as a query for the drug erlotinib (DrugBank ID DB00530) next to the term pancreatic cancer, having a maximum of four words between them (Fig 8). Although this example appears to be slightly artificial, it represents a number of use cases. Boolean queries can increase the precision of the search results, focusing on the word neighbourhood and excluding closely related findings. The usage of such HTML pages with highlighted search

Rank	PubMed-ID	Title (query term highlighted)	Percent
0	16317266	**Erlotinib** and chemoradiation followed by maintenance **erlotinib** for locally advanced **pancreatic cancer**: a phase I study.	100
1	17878488	**Erlotinib** in **pancreatic cancer** patients: do we need more information from the NCIC CTG trial?	96
2	17906218	Does a statistically significant survival benefit of **erlotinib** plus gemcitabine for advanced **pancreatic cancer** translate into clinical significance and value?	96
3	18089885	**Erlotinib** in **pancreatic cancer**: are tumor cells the (only) target?	96
4	18615326	Irreversible ototoxicity associated with the use of **erlotinib** in a patient with **pancreatic cancer**.	96
5	18360654	Role of **erlotinib** in the management of **pancreatic cancer**.	94
6	17704662	Severe lung and skin toxicity during treatment with gemcitabine and **erlotinib** for metastatic **pancreatic cancer**.	93
7	20868602	[Efficacy of gemcitabine combined with **erlotinib** in patients with advanced **pancreatic cancer**].	93
8	17935272	Tyrosine kinase inhibitors in non-small cell lung and **pancreatic cancer**: the emerging role of **erlotinib**.	93

Fig 8. Boolean query result. The HTML page shows a rank in the first column with a relative match score, scaled to 100. The NEAR condition was used to allow up to four other words between the drug erlotinib and the disease term pancreatic cancer without fixed word order.

doi:10.1371/journal.pone.0163794.g008

terms demonstrates a basic way to simplify text curation and annotation. Search results can be ranked with a Xapian-specific score. This illustrates how a full text index can be easily used to display search results in a user-friendly view.

Oliver *et al*. showed a rather complex SQL query, selecting the ten journals that published the most articles with the MeSH term "Leukemia" [20]. Analogously, the slightly modified query to the PubMedPortable database is shown in Table 2. The result is displayed in Table 3, with the outcome that the order of top range journals did not change a lot within the last ten years. In contrast to the other use cases described here, this selection was applied to the complete PubMed data set.

The PubMedPortable documentation on GitHub contains other detailed ideas how to apply the software library, which are not shown here. One example is to process all abstracts and titles which were found with the drug name gemcitabine. The most frequently occurring terms

Table 2. PostgreSQL query to select the ten journals with the highest number of publications containing the MeSH term "Leukemia" [20] on the complete PubMed data set.

SQL command	
SELECT	mj.medline_ta, count(mj.fk_pmid) as num_of_publications
FROM	pubmed.tbl_medline_journal_info mj
JOIN	pubmed.tbl_mesh_heading msh
ON	mj.fk_pmid = msh.fk_pmid
WHERE	msh.descriptor_name = 'Leukemia'
GROUP BY	mj.medline_ta
ORDER BY	count(mj.fk_pmid) desc
FETCH	first 10 rows only;

doi:10.1371/journal.pone.0163794.t002

Table 3. Results for the query shown in Table 2.

Journal	Number of publications
Blood	1,469
Cancer	748
Leukemia	746
Leuk Res	723
Cancer Res	718
Bone Marrow Transplant	710
Br J Haematol	677
Rinsho Ketsueki	671
Lancet	582
Haematologica	486

doi:10.1371/journal.pone.0163794.t003

are shown in an additional word cloud to reveal part of the vocabulary related to pancreatic cancer. A whitespace tokeniser was used that separates words as tokens and removes punctuation. Another example combines a Boolean search in Xapian for selected terms with PostgreSQL queries to identify the main authors by their number of publications. Subsequently, their investigated topics can be compared to the entities in the word clouds. Similar to the query in Table 2, it can be investigated in which countries the most journals in reference to the pancreatic cancer data set are located.

All examples shown here illustrate ideas how PubMedPortable can be applied to user-specific data sets. It depends on the user's aim, what kind of data analysis will be included in a text mining application.

Extending PubMedPortable

The selection of Python, PostgreSQL, and Xapian is based on the requirement to provide a framework that is easy to install, independent from a web service, and usable with only a few lines of code. Using another type of database management system such as Oracle or MySQL with the ORM approach is possible, as well as using the Xapian interfaces for Java or C++ instead of Python. PubMedPortable enables the user to directly create individual queries and to extend this framework with sophisticated NLP approaches. The PubMedPortable database can be used as a centralised repository, which is regularly updated. This also offers the possibility to monitor changes in a data set over time by executing a workflow repeatedly. Furthermore, PubMedPortable supports indexing of unformatted PubMed Central (PMC) full texts. The combination of a relational database and a full text index can also be integrated in a web server, as shown in previous publications [4, 39]. In these projects, the PubMedPortable framework was extended with named entity recognition for gene or protein names from UniProt and PubChem synonyms for small molecules.

Beside the illustrated use case of accessing the MeSH terms in the PubMedPortable relational database, the users might wish to further classify their texts with ontology concepts using the web service of the Open Biomedical Annotator (OBA) [40], e.g. with diseases.

Texts in a corpus can also be categorised using topic models with the Latent Dirichlet allocation as invented by Blei *et al.* [41] and implemented in Python on GitHub by V. Sandulescu.

Another approach is to use tools implemented in a workflow management system like Galaxy [12, 42], supporting drag-and-drop construction of NLP pipelines as applied in the already mentioned Alveo platform [13].

To conclude, the BioC interface provides manifold applications for NLP, but there is also a range of approaches to extract information using other sources as described for some detailed examples. This emphasises that PubMedPortable can be used in a modular way depending on the user's needs.

Conclusion

PubMedPortable offers a ready-to-start solution for developing large-scale text mining applications by generating an in-house database from PubMed articles. The resulting data environment supports complex relational database queries and fast full text search. The BioC interface and the possibility to use Docker provide interoperability to apply NLP approaches in different programming languages and to run queries on several operating systems without much programming effort. PubMedPortable combines and updates selected approaches from related work to result in a state-of-the-art software library as described by the presented use cases. All software and included methods are open-source and free to be modified for further refinements and improvements within the community.

Supporting Information

S1 File. This GitHub file contains the complete PubMedPortable documentation and the source code with all examples and results presented in this article.
(ZIP)

Author Contributions

Conceptualization: BG KD SG.

Data curation: KD SG.

Formal analysis: KD.

Funding acquisition: SG.

Methodology: KD BG PT.

Project administration: SG.

Resources: SG.

Software: KD KT BG.

Supervision: SG.

Visualization: KT KD.

Writing – original draft: KD.

Writing – review & editing: KD PT BG SG.

References

1. Khare R, Leaman R, Lu Z. Accessing Biomedical Literature in the Current Information Landscape. In: Biomedical Literature Mining. vol. 1159. New York, NY: Springer New York; 2014. p. 11–31. doi: 10.1007/978-1-4939-0709-0_2 PMID: 24788259

2. Tikk D, Solt I, Thomas P, Leser U. A detailed error analysis of 13 kernel methods for protein–protein interaction extraction. BMC Bioinformatics. 2013; 14(1):12. doi: 10.1186/1471-2105-14-12 PMID: 23323857
3. Tari L, Anwar S, Liang S, Cai J, Baral C. Discovering drug-drug interactions: a text-mining and reasoning approach based on properties of drug metabolism. Bioinformatics. 2010 Sep; 26(18):i547–i553. doi: 10.1093/bioinformatics/btq382 PMID: 20823320
4. Senger C, Grüning BA, Erxleben A, Döring K, Patel H, Flemming S, et al. Mining and evaluation of molecular relationships in literature. Bioinformatics. 2012 Mar; 28(5):709–714. doi: 10.1093/bioinformatics/bts026 PMID: 22247277
5. Kuhn M, Szklarczyk D, Pletscher-Frankild S, Blicher TH, von Mering C, Jensen LJ, et al. STITCH 4: integration of protein–chemical interactions with user data. Nucleic Acids Research. 2014 Jan; 42(D1): D401–D407. doi: 10.1093/nar/gkt1207 PMID: 24293645
6. Rinaldi F, Clematide S, Marques H, Ellendorff T, Romacker M, Rodriguez-Esteban R. OntoGene web services for biomedical text mining. BMC Bioinformatics. 2014; 15(Suppl 14):S6. doi: 10.1186/1471-2105-15-S14-S6 PMID: 25472638
7. Rak R, Batista-Navarro RT, Carter J, Rowley A, Ananiadou S. Processing biological literature with customizable Web services supporting interoperable formats. Database. 2014 Jul; 2014(0):bau064–bau064. doi: 10.1093/database/bau064 PMID: 25006225
8. Ferrucci D, Lally A. UIMA: an architectural approach to unstructured information processing in the corporate research environment. Natural Language Engineering. 2004 Sep; 10(3-4):327–348. doi: 10.1017/S1351324904003523
9. Cunningham H, Tablan V, Roberts A, Bontcheva K. Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. PLoS Computational Biology. 2013 Feb; 9(2): e1002854. doi: 10.1371/journal.pcbi.1002854 PMID: 23408875
10. Comeau DC, Islamaj Doğan R, Ciccarese P, Cohen KB, Krallinger M, Leitner F, et al. BioC: a minimalist approach to interoperability for biomedical text processing. Database. 2013 Sep; 2013(0):bat064–bat064. doi: 10.1093/database/bat064 PMID: 24048470
11. Kano Y, Baumgartner WA, McCrohon L, Ananiadou S, Cohen KB, Hunter L, et al. U-Compare: share and compare text mining tools with UIMA. Bioinformatics. 2009 Aug; 25(15):1997–1998. doi: 10.1093/bioinformatics/btp289 PMID: 19414535
12. Burnham D, Estival D, Cassidy S, Sefton P, Verspoor K. Two platforms for research in Human Communication Science: The AusTalk corpus and the Alveo Virtual Laboratory. In: Co-ordination and Standardization of Speech Databases and Assessment Techniques (COCOSDA). IEEE; 2014. p. 1–6. doi: 10.1109/ICSDA.2014.7051412
13. Cassidy S, Estival D, Jones T, Burnham D, Burghold J. The Alveo Virtual Laboratory: A Web based Repository API. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14). Reykjavik, Iceland: European Language Resources Association (ELRA); 2014. p. 1–7.
14. Dawborn T, Curran JR. docrep: A lightweight and efficient document representation framework. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. Dublin, Ireland: Dublin City University and Association for Computational Linguistics; 2014. p. 762–771.
15. Khare R, Wei CH, Mao Y, Leaman R, Lu Z. tmBioC: improving interoperability of text-mining tools with BioC. Database. 2014 Jul; 2014(0):bau073–bau073. doi: 10.1093/database/bau073 PMID: 25062914
16. Wei CH, Kao HY, Lu Z. PubTator: a web-based text mining tool for assisting biocuration. Nucleic Acids Research. 2013 Jul; 41(W1):W518–W522. doi: 10.1093/nar/gkt441 PMID: 23703206
17. Sayers E. The E-utilities In-Depth: Parameters, Syntax and More. Entrez Programming Utilities Help, Bethesda (MD): National Center for Biotechnology Information (US); 2015. Available: http://www.ncbi.nlm.nih.gov/books/NBK25499
18. Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics. 2009 Jun; 25 (11):1422–1423. doi: 10.1093/bioinformatics/btp163 PMID: 19304878
19. Alias-i. LingPipe 4.1.0; 2008. Accessed July 20, 2016. Website. Available: http://alias-i.com/lingpipe
20. Oliver DE, Bhalotia G, Schwartz AS, Altman RB, Hearst MA. Tools for loading MEDLINE into a local relational database. BMC Bioinformatics. 2004; 5(1):1–12. doi: 10.1186/1471-2105-5-146 PMID: 15471541
21. Yoo D, Xu I, Berardini TZ, Yon Rhee S, Narayanasamy V, Twigger S. PubSearch and PubFetch: A Simple Management System for Semiautomated Retrieval and Annotation of Biological Information from the Literature. In: Baxevanis AD, Page RDM, Petsko GA, Stein LD, Stormo GD, editors. Current

Protocols in Bioinformatics. Hoboken, NJ, USA: John Wiley & Sons, Inc.; 2006. p. 1–9. doi: 10.1002/0471250953.bi0907s13 PMID: 18428773

22. SimTK. MEDLINE Parser—Load XML MEDLINE Data into RDBMS; 2000. Accessed July 20, 2016. Website. Available: https://simtk.org/home/medlineparser
23. Merkel D. Docker: Lightweight Linux Containers for Consistent Development and Deployment. Linux J. 2014 Mar; 2014(239).
24. Wei CH, Leaman R, Lu Z. Beyond accuracy: creating interoperable and scalable text-mining web services. Bioinformatics. 2016 Jun; 32(12):1907–1910. doi: 10.1093/bioinformatics/btv760 PMID: 26883486
25. Comeau DC, Liu H, Islamaj Do an R, Wilbur WJ. Natural language processing pipelines to annotate BioC collections with an application to the NCBI disease corpus. Database. 2014 Jun; 2014(0): bau056–bau056. doi: 10.1093/database/bau056 PMID: 24935050
26. Garrido-Laguna I, Hidalgo M. Pancreatic cancer: from state-of-the-art treatments to promising novel therapies. Nature Reviews Clinical Oncology. 2015 Mar; 12(6):319–334. doi: 10.1038/nrclinonc.2015.53 PMID: 25824606
27. Huang M, Liu J, Zhu X. GeneTUKit: a software for document-level gene normalization. Bioinformatics. 2011 Apr; 27(7):1032–1033. doi: 10.1093/bioinformatics/btr042 PMID: 21303863
28. Leaman R, Islamaj Dogan R, Lu Z. DNorm: disease name normalization with pairwise learning to rank. Bioinformatics. 2013 Nov; 29(22):2909–2917. doi: 10.1093/bioinformatics/btt474 PMID: 23969135
29. Leaman R, Wei CH, Lu Z. tmChem: a high performance approach for chemical named entity recognition and normalization. Journal of Cheminformatics. 2015; 7(Suppl 1):S3. doi: 10.1186/1758-2946-7-S1-S3 PMID: 25810774
30. Abbasian E, Döring K. GeneTUKit-Pipeline; 2015. Accessed July 20, 2016. Website. Available: https://github.com/ElhamAbbasian/GeneTUKit-Pipeline
31. Amberger J, Bocchini CA, Scott AF, Hamosh A. McKusick's Online Mendelian Inheritance in Man (OMIM(R)). Nucleic Acids Research. 2009 Jan; 37(Database):D793–D796. doi: 10.1093/nar/gkn665 PMID: 18842627
32. Law V, Knox C, Djoumbou Y, Jewison T, Guo AC, Liu Y, et al. DrugBank 4.0: shedding new light on drug metabolism. Nucleic Acids Research. 2014 Jan; 42(D1):D1091–D1097. doi: 10.1093/nar/gkt1068 PMID: 24203711
33. Fernandez-Cruz L. Surgical treatment: evidence-based and problem-oriented. Zuckschwerdt; 2001. Available: http://www.ncbi.nlm.nih.gov/books/NBK6924
34. Ryan DP, Hong TS, Bardeesy N. Pancreatic Adenocarcinoma. New England Journal of Medicine. 2014 Sep; 371(11):1039–1049. doi: 10.1056/NEJMra1404198 PMID: 25207767
35. Döring K. Workflows to predict functional relationships of compounds and proteins in texts by using the all-paths graph kernel and the shallow linguistic kernel; 2016. Accessed July 20, 2016. Website. Available: https://github.com/KerstenDoering/CPI-Pipeline
36. Comeau DC, Batista-Navarro RT, Dai HJ, Islamaj Do an R, Jimeno Yepes A, Khare R, et al. BioC interoperability track overview. Database. 2014 Jun; 2014(0):bau053–bau053. doi: 10.1093/database/bau053 PMID: 24980129
37. Proceedings of the Fifth BioCreative Challenge Evaluation Workshop, Seville, Spain; 2015. Available: http://www.biocreative.org/resources/biocreative-v/proceedings-biocreative
38. Zimmermann G, Papke B, Ismail S, Vartak N, Chandra A, Hoffmann M, et al. Small molecule inhibition of the KRAS–PDEδ interaction impairs oncogenic KRAS signalling. Nature. 2013 May; 497(7451):638–642. doi: 10.1038/nature12205 PMID: 23698361
39. Grüning BA, Senger C, Erxleben A, Flemming S, Günther S. Compounds In Literature (CIL): screening for compounds and relatives in PubMed. Bioinformatics. 2011 May; 27(9):1341–1342. doi: 10.1093/bioinformatics/btr130 PMID: 21414988
40. Jonquet C, Shah NH, Musen MA. The Open Biomedical Annotator. Summit on Translational Bioinformatics. 2009; 2009:56–60. PMID: 21347171
41. Blei DM, Ng AY, Jordan MI. Latent Dirichlet Allocation. J Mach Learn Res. 2003 Mar; 3(4-5):993–1022.
42. Goecks J, Nekrutenko A, Taylor J, Galaxy Team T. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biology. 2010; 11(8):R86. doi: 10.1186/gb-2010-11-8-r86 PMID: 20738864