Journal of the American Medical Informatics Association, 28(3), 2021, 504–515
doi: 10.1093/jamia/ocaa261
Advance Access Publication Date: 15 December 2020
Research and Applications

Research and Applications

Can reproducibility be improved in clinical natural language processing? A study of 7 clinical NLP suites

William Digan[1,2], Aurélie Névéol[3], Antoine Neuraz[1,4], Maxime Wack[1,2], David Baudoin[1,2], Anita Burgun[1,2,4] and Bastien Rance[1,2]

[1]INSERM, Centre de Recherche des Cordeliers, UMRS 1138, Université de Paris, Université Sorbonne Paris Cité, Paris, France, [2]Department of Medical Informatics, Hôpital Européen Georges Pompidou, Assistance publique–Hôpitaux de Paris, Paris, France, [3]Université Paris Saclay, CNRS, LIMSI, Orsay, France and [4]Department of Medical Informatics, Necker Children's Hospital, Assistance publique–Hôpitaux de Paris, Paris, France

Corresponding Author: Bastien Rance, European Hospital Georges Pompidou, 20 rue Leblanc, 75015 Paris France (bastien.rance@aphp.fr)

Received 7 July 2020; Editorial Decision 5 October 2020

ABSTRACT

Background: The increasing complexity of data streams and computational processes in modern clinical health information systems makes reproducibility challenging. Clinical natural language processing (NLP) pipelines are routinely leveraged for the secondary use of data. Workflow management systems (WMS) have been widely used in bioinformatics to handle the reproducibility bottleneck.
Objective: To evaluate if WMS and other bioinformatics practices could impact the reproducibility of clinical NLP frameworks.
Materials and Methods: Based on the literature across multiple researcho fields (NLP, bioinformatics and clinical informatics) we selected articles which (1) review reproducibility practices and (2) highlight a set of rules or guidelines to ensure tool or pipeline reproducibility. We aggregate insight from the literature to define reproducibility recommendations. Finally, we assess the compliance of 7 NLP frameworks to the recommendations.
Results: We identified 40 reproducibility features from 8 selected articles. Frameworks based on WMS match more than 50% of features (26 features for LAPPS Grid, 22 features for OpenMinted) compared to 18 features for current clinical NLP framework (cTakes, CLAMP) and 17 features for GATE, ScispaCy, and Textflows.
Discussion: 34 recommendations are endorsed by at least 2 articles from our selection. Overall, 15 features were adopted by every NLP Framework. Nevertheless, frameworks based on WMS had a better compliance with the features.
Conclusion: NLP frameworks could benefit from lessons learned from the bioinformatics field (eg, public repositories of curated tools and workflows or use of containers for shareability) to enhance the reproducibility in a clinical setting.

Key words: reproducibility of results, natural language processing, meaningful use, workflow, containerization

INTRODUCTION

Reproducing experiments is a core scientific activity that forms the basis of advancing knowledge. A recent survey in many fields of science has shown that reproducing results is challenging for researchers, whether it is for their own work or others'.[1] The term *reproducibility* has been widely used in the literature to refer to dif-

ferent activities (sometimes also called *replicability* or *repeatability*). One such activity is attempting to reiterate previously conducted experiments using an identical experimental setup with the goal of assessing the technical soundness of the process. A different activity consists in running a previously conducted experiment with intended variations in the materials (eg, a corpus) or methods (eg, the type of preprocessing applied to that corpus) used with the goal of assessing the impact of the variations induced and providing information about the robustness and generalizability of the algorithm. The different categories of experiment reiteration have been articulated to incorporate increasing levels of variation to the original experiments as *repeat*, *replicate*, *reproduce*, *reuse*,[2] and we will use this terminology throughout the article. The reiteration of computational experiments is subject to many challenges including access to experimental materials,[3,4] computational dependencies, code robustness and software quality.[5,6] Herein we study the requirements for the efficient reiteration of previously conducted experiments in a setting that strives to be identical—or as close as possible—as the original experimental setting in order to facilitate the replication and reproducibility of experiments. We also investigate how modern clinical natural language processing (NLP) suites meet these requirements.

In the field of NLP, Pederson drew the attention of the community to the obstacles to reproducibility almost 10 years ago with the tale of the *Zigglebottom tagger*,[7] which illustrates a typical chain of circumstances that makes reproducibility difficult to the dismay of the researchers involved. Others[8] then built on a failed reproducibility study to pinpoint specific steps that hinder reproducibility in an NLP pipeline. They showed that preprocessing steps such as tokenization are not described in detail although they can have a significant impact downstream in the NLP pipeline. [9] further characterized reproducibility by introducing the 3 dimensions of value (a number, whether measured or calculated, obtained as part of the experiment), finding (a relationship between experimental values), and conclusion (a broad induction based on the results of the experiments). They showed that reproducibility can legitimately apply to a subset of these dimensions only. For example, in the case of nondeterministic algorithms, for a particular iteration, a *value* could not be reproduced while the *finding* (value A is greater than value B) could be reproduced. Furthermore, reproducing a *conclusion* from the same *finding* is not systematic if subjective judgement is involved in the analysis of the result. In a recent editorial,[10] the Journal of the American Medical Informatics Association editor-in-chief, listed 4 categories that need to be clarified in publications to facilitate reproducibility (ie, data, code, connect, and publish). In particular, data (eg, record how each result was produced) and code (eg, record all external programs used) are presented as very important and were previously found to be insufficiently reported in clinical NLP studies.[11]

Indeed, the issue of reproducibility for NLP is critical in a clinical setting, where the quality of the NLP processes (eg, for information extraction and concept normalization) may directly impact the secondary use of the extracted data (eg, the results of public health studies).[12] Subsequently, health policies supported by the results of public health studies relying on this data may also be misled.

The field of bioinformatics[13] is well acquainted with similar issues around reproducibility. Four levels of reproducibility have been articulated by[2] as: *Repeat*, when an experiment can be performed again on the same computational environment and yield the same results; *Replicate*, when the experiment can be performed in a new setting and still yield the same results; *Reproduce*, when an experiment validates the same scientific hypothesis by using a different method, or a different dataset; *Reuse*, when a different experiment is performed with similarities to the original experiment (eg, the same methods are applied to a range of datasets and support the original conclusion). Over the years, methods and tools[14,15] have been developed to handle reproducibility efficiently. In particular, the use of workflow management systems (WMS) has had a great impact and has become virtually ubiquitous in "production-stage"[2] analyses. Nextflow,[16] SnakeMake,[17] and Galaxy[18] are the most used workflow management systems within the bioinformatics community. A WMS is composed of a set of tools and an "orchestrator" designed to build and run pipelines. A pipeline consists of a sequence of different tools, possibly combined in a nonlinear fashion. Within a WMS, each tool is defined by specific inputs and outputs and is used as a black box to perform its function. A WMS helps design how tools will interact through a stream of input and output data. Furthermore, WMS often allow the use of *container technology* (eg, Docker,[19] LXC,[20] Singularity[21]), a method to encapsulate and isolate an application that makes it independent from the operating system. As a result, different versions of the same tools or libraries can coexist in different containers without interference from the host system. Finally, WMS are intertwined with the notion of provenance. *Provenance* is an important dimension of reproducibility. Provenance keeps track of the pipelines used to generate a result, of the elements involved in the process, and so forth. A large body of literature has been dedicated to provenance of data[22–24] and, more specifically, to provenance of biomedical data generated using WMSs.[25] The World Wide Web Consortium (W3C) ontology PROV[26] is largely used to express the provenance of data. In addition to bioinformatics applications, some NLP pipelines use WMS.[27,28]

The general goal of clinical NLP is to transfer the results of NLP research into clinical practice through the *reuse* of solutions (ie, the deployment of tools and models that are successful in 1 clinical environment into another environment and characterized by its own informatics setup and clinical data). The initial step towards this deployment is the ability to effectively replicate workflows with identical tools, datasets, and outcomes. In this study, we focus our interest on clinical NLP deployed in the hospital and the dimensions of reproducibility associated with the production of data (and not secondary use). We identify a need for actionable reproducibility from the inception of experiments to the final results integrated in clinical practice. With the advent of text use in clinical artificial intelligence algorithms, we must be in a position to assess the impact of changes anywhere in the processing pipeline to provide supporting evidence for algorithm outputs and to facilitate hospital interoperability. We do not seek to invent new formalisms but to integrate existing definitions and adapt them to our specific use cases. To limit the scope, this research focuses on methods to ensure technical reproducibility and facilitate code maintenance and sharing. To achieve this goal, data sharing as outlined in the Findability, Accessibility, Interoperability, and Reusability of digital assets (FAIR) principles is a key element.[3] In addition, we emphasize the need to replicate, reproduce and, finally, reuse code in a production clinical environment.

The contributions of this study are threefold. First, we explored the literature in the fields of natural language processing, medical informatics, and bioinformatics to identify technical basic reproducibility features for clinical NLP pipelines. Second, we analyzed popular NLP frameworks which implement or not a WMS in light of selected reproducibility features. Finally, we explored how the recommendations could be ranked and used to consolidate the reproducibility of tools and workflows.

MATERIALS AND METHODS

Identification of articles discussing reproducibility and NLP frameworks

Exploration of literature on reproducibility and WMS

We explored the literature in the fields of NLP, medical informatics, and bioinformatics that discussed reproducibility as a major topic. We searched the literature using PubMed and Web of Science for articles ranging from Jan. 2010 to Dec. 2019. We designed 2 queries. The first query focused on identifying bioinformatics articles which mentioned at least the word *workflow* and other key terms. The second query aimed at identifying NLP or clinical NLP articles. As a complement, the expertise of authors (AB, AN, BR, and WD) was used to yield an initial set of articles, and we then used snowballing based on this set; that is, we checked references cited in the selected articles for possible inclusion.

Table 1 presents the queries used to explore the literature. The second line presents the query used to yield bioinformatics reproducibility articles, the third line presents the query used to yield NLP or clinical NLP reproducibility features, and the final line presents the query used for the identification of the NLP framework.

We included articles in our study if they discussed reproducibility features or NLP frameworks.

Table 1. Literature queries used to identify articles related to reproducibility

Topics	Query
Bioinformatics reproducibility articles	*(workflow management system OR computational workflow OR Scientific workflow OR interoperable workflow OR scalable workflow OR sharing workflow OR Workflow) AND (computational reproducibility OR computational replicability OR computational repeatability OR computational replication or computational reproducible)*
NLP or clinical NLP reproducibility features	*(reproducibility OR replicability OR repeatability OR replication OR reproducible OR transparency) AND ("EHR" OR "biomedical NLP" OR "clinical NLP" OR "clinical natural language processing" OR "natural language processing" OR "NLP" OR "natural language processing")*
Identification of NLP framework	*("EHR" OR "biomedical NLP" OR "clinical NLP" OR "clinical natural language processing" OR "natural language processing" OR "NLP" OR "natural language processing") AND ("system" OR "framework" OR "tool" OR "workflow" OR "pipeline" OR "architecture") AND ("text mining" OR "platform")*

Abbreviations: EHR, electronic health record; NLP, natural language processing.

Identification of reproducibility articles

The criteria for selecting the articles were: (1) review reproducibility practices and articles in the target research field (2) highlight a set of rules or guidelines for tools or pipelines. According to the topics addressed, articles are tagged either as a tool or WMS. To be specific, articles were categorized as WMS if they discussed WMS, pipeline, or strategy to use multiple tools at once (within a framework). By default, other articles describing a piece of software which processes input data and produces output data are categorized as tools.

Identification of NLP frameworks

Using the query build in the previous section, we screened the results for NLP frameworks, that is, systems that (i) are able to run NLP tools, (ii) allow the customization of NLP pipelines, (iii) can handle additional tools, and (iv) deploy freely in a clinical setting.

PRISMA-like flowcharts[29] describing the selection process are available as Supplementary Material (Supplementary Table 1: Reproducibility literature and Supplementary Table 2: NLP frameworks).

Definition of technical reproducibility recommendations

[1,9]draw the baseline of technical reproducibility requirements for the candidate recommendations as a set of questions: (1) Does it facilitate code reuse? (2) Does it ease code maintenance? (3) Does it facilitate customization and local tailoring? (4) Does it provide some experimental context (CPU usage)? (5) Can anybody rerun an experiment and obtain the same results, anytime? We expanded those questions and modulated them from our experience deploying NLP tools in a clinical setting for a language other than English (namely French). We want to pinpoint that tools and workflows impact reproducibility at different granularity and in different ways.

Level of application of the recommendations

Some recommendations are dedicated solely to tools or to workflow. Within our literature review, some articles[9,10,15,30,31] mainly focus on tools, whereas[2,13,24] focused on pipelines management within WMS. We organized the recommendations based on the main target of reproducibility (tool or workflow).

RESULTS

Identification of articles discussing reproducibility

We identified 455 articles using Web of Science and PubMed. After filtering first on titles, then on abstracts, and using full texts for snowballing, we identified 8 articles of interest.[2,9,10,13,15,24,30,31] Figure 1 illustrates the distributions of the selected articles over the spectrum of research fields and according to the level of analysis proposed in the work (foundational or application) and scope category (tools or WMS).

A framework to analyze reproducibility

Table 2 presents the detailed results of our analysis, including a comprehensive list of features tracked back to the articles that define them. Additionally, we leveraged our experience deploying NLP tools in a clinical setting for a language other than English (ie, French). We identified 5 topics of reproducibility (namely, traceability, versioning, standardization, usability, and shareability) detailed in the section below covering 40 recommendations (including "Archived tools version" and "used standard input format").

Traceability describes the ability of the framework to record the context of an experiment and all the environment metadata. It covers information such as provenance (how a result was obtained) and system metadata (including configuration parameters). *Standardization* covers a variety of domains including interoperability through the use of standard (input, output, or resources). Furthermore, we focus on technical choices which will reduce the burden of local adaptation (tools or pipelines). *Versioning* deals with the necessity of keeping under version control each element of a pipeline. *Usability* was divided into 2 categories: (1) automation: includes the absence of manual steps, the ability to scale up, and the ability to resume a

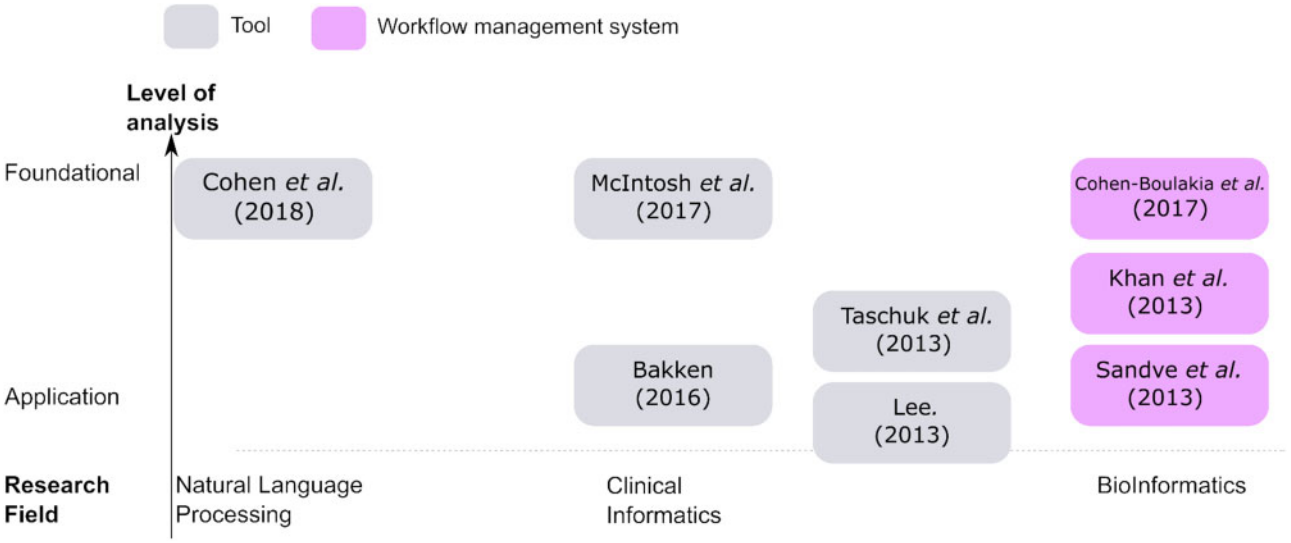

Figure 1. Reproducibility articles sorted by level of analysis and research fields. The scope category as either tool or WMS is also shown.

workflow run after interruption; (2) simplicity of use. These highlight aspects of workflow flexibility (eg, customization of the workflow, management of multiple programming languages). *Shareability* describes the availability of tools, data, WMS, and resources for the community. Recommendations can be applied at the tool or workflow level. The details of the results are illustrated in Figure 2.

Overview of NLP platforms and frameworks

We identified 11 191 articles using Web of Science and PubMed. After filtering first on titles, then on abstracts, and using full texts for snowballing, we describe the 7 selected NLP frameworks (cTakes,[32] CLAMP,[33] GATE,[34] ScispaCy,[35] TextFlow,[36] LAPPS Grid,[27] and OpenMinTed[28]) This selection comprises state-of-the-art NLP platforms (cTakes, CLAMP, GATE) and modularly designed platforms (TextFlow, LAPPS Grid, OpenMinted), which all appeared multiple times through the query. In addition, an expert-recommended tool used in the NLP community and adapted to the biomedical domain was analyzed to increase the diversity of framework profiles (ScispaCy). Table 3 provides basic information related to the frameworks.

cTakes, Clinical Text Analysis, and Knowledge Extraction System (2010)

cTakes[32] is an NLP open source solution used through command line and graphical interfaces. cTakes relies on the UIMA model and is released with a default clinical NLP pipeline. Currently, cTakes is deployed across many American hospitals and universities. cTakes is largely referenced in the literature and is a de facto baseline in many English language NLP experiments. The latest stable release is the version 4.0.0 from April 25, 2017, but cTakes is still under development. cTakes performs many NLP operations ranging from tokenization to NER with a subset of Unified Medical Language System (UMLS). This subset contains RxNorm and SNOMED CT terms and can be tailored by the user.

CLAMP, Clinical Language Annotation Modelling and Processing (2017)

CLAMP[33] is currently in use by 509 organizations. CLAMP is a graphical user interface or command line clinical NLP solution which relies on the UIMA model. The latest release 1.4.0 is freely available for research use upon submission of a request form. CLAMP architecture is based on 3 pillars:[38] First, the NLP Pipeline components allow the creation of a pipeline by a simple drag and drop of CLAMP NLP components; second, a set of NLP pipelines is supplied for common clinical applications (eg, extraction of smoking status, colorectal cancer, etc). The machine learning and hybrid approach pillar supplies alternative methods (eg, support vector machine, rule-based methods, etc) which can be retrained on a new corpus. The third pillar is a corpus management and annotation tool that provides a built-in text annotation interface based on brat.[39]

GATE, General Architecture for Text Engineering (1995)

GATE[34] is a multilingual text engineering platform. GATE was developed 25 years ago and has been continuously updated since then. GATE is built around the purposes of easing successful and sustainable deployment, being scalable, and reusing existing tools. GATE comes with a suite of applications. The GATE developer is an interface development environment in which users can process text from a set of graphical interactive tools. Processes and pipelines that are built can be sent to GATE Cloud or GATE Teamware. GATE Embedded provides an optimized object library and access to all GATE services. GATE Teamware is a web-based collaborative annotation platform. GATE Mimir and GATE Cloud are designed to perform scalable and parallel analysis. Moreover, GATE integrates existing tools, such as LingPipe[40] or OpenNLP.[41]

ScispaCy, spaCy pipeline, and models for scientific documents (2019)

ScispaCy[35] is based on spaCy[42] but is dedicated to scientific projects. ScispaCy proposed 4 named entity recognition (NER) models dedicated to English trained on 4 different corpora, including a biomedical one (BIONLP13CG). ScispaCy also retrained 3 spaCy mod-

Table 2. Characterization of reproducibility recommendations collected from the literature. Each recommendation is assigned a topic and a simple description. The coverage of the recommendation in a given article is noted as present (✓) or absent (-)

Topics	ID	Features	24	2	9	10	30	13	15	31
Traceability	R01	Provenance metadata	✓	✓	–	✓	✓	–	–	–
	R02	Generating execution logs	✓	✓	–	✓	–	✓	✓	✓
	R03	System metadata (eg, RAM, CPU, OS, etc)	✓	✓	–	–	–	–	–	–
	R04	Record parameters of tools	✓	✓	–	–	–	✓	–	✓
	R05	Recording intermediate results	✓	✓	–	✓	✓	✓	–	–
Versioning	R06	Use of version control for workflows	✓	✓	✓	✓	–	✓	–	✓
	R07	Use of version control for tools	✓	✓	✓	✓	–	✓	–	✓
	R08	Use of version control for resources	✓	✓	✓	✓	–	✓	–	✓
	R09	Archived tools versions	✓	✓	–	✓	✓	✓	–	✓
	R10	Archived WMS versions	–	–	–	✓	–	✓	–	✓
	R11	Archived resources versions	–	–	–	✓	–	✓	–	✓
	R12	Archived input data versions	✓	–	–	✓	–	✓	–	✓
Standardization	R13	Standard objects identifier input data	✓	–	–	✓	✓	✓	–	–
	R14	Standard objects identifier output data	✓	–	–	✓	✓	✓	–	–
	R15	Standard objects identifier resources	✓	–	–	✓	✓	✓	–	–
	R16	Standard objects identifier tools	✓	–	–	✓	✓	✓	–	–
	R17	Use of research objects	✓	✓	–	–	–	–	✓	✓
	R18	Containerization	✓	✓	–	–	–	–	–	✓
	R19	Use of relative path within tools/ - hard coded path	✓	–	✓	–	–	–	–	✓
	R20	Use of standard folder organization at workflow level (eg, BagIt)	✓	–	–	–	–	–	–	–
	R21	Use of standard folder organization at tools level (eg, BagIt, Django project, Eclipse plugin, etc)	✓	–	–	–	–	–	–	–
	R22	Presence of a README (tool)	–	–	–	–	✓	–	✓	✓
	R23	Presence of a README (workflow)	–	–	–	–	✓	–	✓	✓
	R24	Availability of a full documentation	–	–	–	–	✓	–	✓	✓
	R25	Tools use a standard framework (eg, UIMA, Docker)	–	–	–	–	–	–	✓	–
	R26	Input data in a standard format (eg, BioC, JsonNLP)	✓	–	–	–	–	–	✓	–
	R27	Output data in a standard format (eg, BioC, JsonNLP)	✓	–	–	–	–	–	✓	–
Usability	R28	Absence of manual steps	✓	–	–	✓	–	✓	–	✓
	R29	Ability to scale up	✓	–	–	–	–	–	–	✓
	R30	Ability to resume a workflow run	✓	✓	–	–	–	–	–	–
	R31	Ability to customize the workflow	✓	–	–	–	–	–	–	–
	R32	Management of multiple programming languages	–	–	–	–	–	–	–	–
	R33	Workflow Modularity (use or share parts of the workflow)	✓	✓	–	–	–	–	–	–
	R34	Licensing	✓	–	–	–	✓	–	–	✓
	R35	Benchmark data and performance distributed with the tools	✓	–	–	–	–	–	✓	✓
	R36	Identification of tools to be tailored locally (eg, preprocessing, local rules)	–	–	–	–	✓	–	–	–
Shareability	R37	Workflow publicly accessible	✓	✓	✓	✓	✓	✓	✓	–
	R38	Tools publicly accessible	✓	✓	✓	✓	✓	✓	✓	–
	R39	Input data publicly accessible	✓	–	✓	✓	✓	✓	✓	–
	R40	Resources publicly accessible	–	–	✓	–	–	✓	✓	–

els for part-of-speech (POS) tagging, dependency parsing, rule-based tokenizer, and syntactic parser trained on biomedical datasets. ScispaCy also proposed additional pipeline components such as AbbreviationDetector and an EntityLinker linked to the UMLS, RxNorm, GeneOntology, Medical Subject Headings (MeSH), and the Human Phenotype Ontology. ScispaCy leverages external libraries, such as tensorflow, for deep learning models. ScispaCy (similarly to spaCy) relies on coding skills (ie, not on graphical user interface). While resources for languages other than English are available in spaCy, ScispaCy does not offer resources outside of English.

TextFlows, text mining workflows platform (2015)

TextFlows,[36] a fork of ClowdFlows,[43] is an open source web interface and focuses on smoothing the construction, execution, and shareability of NLP workflows through a web interface. TextFlows was not designed for clinical narratives, and its last update was December 1, 2017. TextFlows NLP tools are called "widgets" (eg, lemmatization or POS) and are assembled to produce workflows. Text mining widgets embed several software packages (eg, NLTK[44] and scikit-learn[45]) and cover a wide range of NLP analysis (eg, tokenization, stop word removal, POS tagging, stemming, and lemmatization). Moreover, users can extend TextFlows by adding their own widget. TextFlows is developed as a Django Python web interface and can be run on a cluster of servers.

OpenMinTed, Open Mining Infrastructure for TExt and Data (2015)

OpenMinTed[28] emphasized the use of text and data mining in the context of open access scientific publication. OpenMinted is built around 3 building blocks. The first is named "Content resources" and represents open access articles used as input for text and data mining software. The second building block, "Text and data mining software," performs basic NLP tasks (eg, tokenization, sentence splitting, NER, etc) split between component and applications. The

Figure 2. Classification of recommendation at tool and pipeline level. 21 recommendations are applicable both to tools and workflows, 12 to workflows only, and 7 to tools only.

last building block, "Ancillary knowledge resources," supports the management of ontologies, terminologies or machine learning models. Furthermore, OpenMinTed dispenses OMTD-SHARE, a metadata schema which covers description of resources and text and data-mining software to enhance interoperability. OpenMinTed is developed in Java; workflows are executed within Galaxy.

Grid galaxy, the language application grid, and galaxy flavor (2016)
LAPPS Grid[27] aspired to ease the creation of workflows from hundreds of NLP tools indexed in a library and facilitate the sharing of data. The LAPPS Grid Project is a web interface which contains a wide range of NLP tools (eg, sentence splitters, tokenizers, etc). Tools came from multiple providers, such as Stanford CoreNLP,[46] Apache open NLP,[47] and Gate.[37] The LAPPS Grid Project released a Galaxy *flavor* called Grid Galaxy in March/April 2018, available as Docker containers.

Table 4 illustrates the adherence of the frameworks to the reproducibility features.

DISCUSSION

Reproducibility recommendations are articulated across all research fields

We elaborated the recommendations based on the 8 articles studying reproducibility in the light of our daily experience of the deployment and use of NLP processes in a clinical setting. As illustrated in Table 2, most of the features (39/40) are endorsed by at least 1 article in our selection. 31 recommendations are cover by[24]. However, it can be noted that the granularity of the recommendations in[24]'s provided by the Common Workflow Language Project differs from ours. In addition, some recommendations from our list are discussed by[24] but not selected as part of their recommendation list. Recommendations R25, R26, R27 (addressing standardization), R31, and R36 (addressing usability) are found only in a single article. These findings are mainly on par with our experience. We enriched the list with 1 feature our own: R32, *management of multiple programming languages*. This overview shows that the key features of reproducibility have been articulated by the different communities.

NLP frameworks implement an incomplete set of reproducibility features

Selection and classification of NLP frameworks

The major selection criteria were (i) the deployment capability of the NLP framework in a clinical setting and (ii) the ability to process large amounts of data. Identified NLP frameworks are regrouped in 3 categories: (a) the ones relying on the UIMA[29] framework (namely cTakes and CLAMP) or legacy system (GATE); (b) the systems relying on WMS, such as Galaxy[10] (LAPPGrid, OpenMinTed) or following WMS principle (Textflows); ScispaCy belongs to a third category, (c) NLP toolkit. NLP toolkits are often supposed to be integrated into complex NLP pipeline programmatically.

Need to enhance versioning, standardization, and shareability across NLP frameworks

Overall, all frameworks shared 11 features of reproducibility across traceability (R2, R4, R5), standardization (R13, R21, R22, R24), and usability (R28, R29, R31, R34) with different levels of formalism or maturity. Most of the frameworks are linked to an open source git repository and implement at least 1 shareability recommendation. Nevertheless, none of the systems allow a control of the versions of tools and pipelines. That is, a user cannot change a specific tool to a specific version and only has access to ready-to-use builds of the entire system.

Looking at traceability, each framework provided some types of provenance metadata. However, none of them natively adopted the PROV and PROV-O ontology. Instead the systems relied on *ad hoc*

Table 3. Summary information on the NLP frameworks selected for our study

	cTakes[32]	CLAMP[33]	GATE[37]	ScispaCy[35]	TextFlows[36]	OpenMinteD[28]	Grid Galaxy[27]
project initiators	Mayo Clinic	School of Biomedical Informatics at the University of Texas Health at Houston	The University of Sheffield, South Yorkshire, England	Allen Institute for Artificial Intelligence, Seattle, WA, USA	Jožef Stefan Institute, Ljubljana, Slovenia	Athena Research and Innovation Center in Information, Communication and Knowledge Technologies	Vassar College, Poughkeepsie, NY USA Brandeis University, Vassar College, Carnegie-Mellon
availability	Open source	Upon request	Open source	Open source	Open source	Open source	Open source
Licensing	Apache	/	LGPL-3.0	Apache	MIT	Apache	Apache
language	Java	Java	Java	python	python	Python/Java	Python/ Java
Source	https://github.com/apache/ctakes	https://clamp.uth.edu/getclamp.php	https://github.com/GateNLP/gate-core	https://github.com/allenai/scispacy	https://github.com/xflows/textflows	https://github.com/openminted/registry/	https://github.com/lappsgrid-incubator/galaxy-appliance
Demo	https://ctakes.apache.org/downloads.cgi	https://clamp.uth.edu/clampdemo.php	https://gate.ac.uk/demos/	https://scispacy.apps.allenai.org/	http://textflows.org/	https://github.com/openminted/install-tutorial	https://galaxy.lappsgrid.org/
Tools format	UIMA	UIMA	GATE	scpaCy	widget	Galaxy tool	Galaxy tool
Container	https://github.com/tmills/ctakes-docker https://github.com/choyiny/ctakes-server		https://github.com/CogStack/gate-nlp-service	https://github.com/allenai/scispacy		https://github.com/openminted/omtd-docker https://github.com/openminted/omtd-docker-specification	https://hub.docker.com/r/lappsgrid/galaxy

Table 4. NLP frameworks described in the frame of reproducibility recommendations: we assign the presence (✓) partial (/) or absence or no information (-) of each recommendation in NLP frameworks

	Features	cTakes[32]	CLAMP[33]	GATE	ScispaCy	TextFlows[36]	OpenMinteD[28]	LAPPS Grid Galaxy[27]
R01	Provenance metadata	✓	–	–	–	✓	✓	✓
R02	Generating execution logs	✓	✓	✓	✓	✓	✓	✓
R03	System metadata (eg, RAM, CPU, OS, etc)	✓	✓	✓	–	✓	✓	✓
R04	Record parameters of tools	✓	✓	✓	✓	✓	✓	✓
R05	Recording intermediate results	✓	✓	✓	✓	✓	✓	✓
R06	Use of version control for workflows	–	–	–	–	–	–	–
R07	Use of version control for tools	–	–	–	–	–	–	–
R08	Use of version control for resources	–	–	–	–	–	–	–
R09	Archived tools versions		–	–	–	–	–	–
R10	Archived WMS versions	–	–	–	–	–	–	–
R11	Archived resources versions	–	–	–	–	–	–	–
R12	Archived input data versions	–	–	–	–	–	–	–
R13	Standard objects identifier input data	✓	✓	✓	✓	✓	✓	✓
R14	Standard objects identifier output data	✓	✓	–	✓	✓	✓	✓
R15	Standard objects identifier resources	–	–	✓	–	✓	✓	✓
R16	Standard objects identifier tools	–	–	✓	–	✓	✓	✓
R17	Use of research objects	–	–	–	–	–	–	–
R18	Containerization	–	–	–	–	–	✓	–
R19	Use of relative path within tools/—hard coded path	–	–	–	–	–	–	–
R20	Use of standard folder organization at workflow level (eg, BagIt)	–	–	–	–	–	✓	✓
R21	Use of standard folder organization at tools level (eg, BagIt, Django project, eclipse plugin, etc)	✓	✓	✓	✓	✓	✓	✓
R22	Presence of a README (tool)	✓	✓	✓	✓	✓	✓	✓
R23	Presence of a README (workflow)	✓	✓	–	–	–	–	✓
R24	Availability of a full documentation	✓	✓	✓	✓	✓	✓	✓
R25	Tools use a standard framework (eg, UIMA, Docker, Galaxy)	✓	✓	✓	–	–	–	✓
R26	Input data in a standard format (eg, BioC, JsonNLP, LIF)	/	–	–	–	–	–	✓
R27	Output data in a standard format (eg, BioC, JsonNLP, LIF)	/	–	–	–	–	–	✓
R28	Absence of manual steps	✓	✓	✓	✓	✓	✓	✓
R29	Ability to scale up	✓	✓	✓	✓	✓	✓	✓
R30	Ability to resume a workflow run (after failure)	–	–	–	✓	–	✓	✓
R31	Ability to customize the workflow	✓	✓	✓	✓	✓	✓	✓
R32	Management of multiple programming languages	–	–	–	–	–	✓	✓
R33	Workflow modularity (use or share parts of the workflow)	–	–	–	–	–	✓	✓
R34	Licensing	✓	✓	✓	✓	✓	✓	✓
R35	Benchmark data and performance distributed with the tools	✓	✓	✓	✓	–	–	–
R36	Identification of tools to be tailored locally (eg, pre-processing, local rules)	–	–	–	–	–	–	–
R37	Workflow publicly accessible	✓	✓	–	✓	✓	✓	✓
R38	Tools publicly accessible	–	–	✓	✓	–	/	✓
R39	Input data publicly accessible	–	–	–	–	–	–	–
R40	Resources publicly accessible	–	–	–	✓	–	–	–

metadata. It is worth noting that the community of users has made an attempt to enrich the system. For example,[30] have proposed ProvCare. OpenMinTed has its own metadata schema called OMTD-SHARE.

Standardization of data interaction through the Galaxy framework aims at ensuring replicability and reproducibility of analysis. As an example, each datum uploaded in Galaxy is associated to an identifier and a type (BAM, FastQ, etc). Furthermore, each datum produced by a tool is also associated with the unique identifier of the tool. Note that recommendation R13 to R16 (URI and standard vocabulary), are also recommended in the FAIR principles. LAPPS Grid fully integrated their pipeline in Galaxy and provided to the community a Galaxy flavor as a container solution. Also, OpenMinTeD pipelines are executed with Galaxy. Furthermore, Galaxy can handle tools encapsulated in containers such as Docker.[19] NLP frameworks currently do not support container tools integration. Overall, the framework that meets the most recommendations is LAPPS Grid (N = 26), while GATE, ScispaCy, and Textflows meet

the least (N = 17). cTakes and CLAMP meet 18 recommendations. This was to be expected given that LAPPS Grid is implemented directly within Galaxy WMS, which explains the highest compliance.

Remarks on UIMA in NLP frameworks

The UIMA standard architecture[48] is used in several NLP frameworks (eg, cTakes, CLAMP, etc). UIMA largely relies on the Java programming language, although new implementations are slowly made available in other languages. Similar to the field of bioinformatics, NLP frameworks should allow the combination of tools written in a variety of programming languages. Not all tools will be developed in Java, especially with the rise of Python (used in ScispaCy) as a data-science language. The use of standards, ontologies, and WMS could help the development of sharable and modular tools. This seems especially crucial in languages other than English, for which the resources are scarce, and the adaptation of existing pipelines is prohibitively costly.

Remarks on clinical data sharing

Clinical corpus sharing is restricted for confidentiality reasons, and FAIR principles cannot always be applied easily. In many cases, corpora cannot be shared outside the hospital that hosts them. As a result, shared datasets are limited to American English narratives (eg, based on MIMIC[49]) or substitutes to clinical corpora, such as clinical case descriptions (eg, https://temu.bsc.es/cantemist/[50]), MEDLINE biomedical literature (eg,[51]), or micro blogs (eg, SSM4H[52]). Data access issues include the attrition or modification of dataset content over time when data elements (such as twitter posts and MEDLINE citations) are distributed in the form of identifiers. NLP researchers and tool developers are bound by data sharing constraints. However, these limitations should be taken into account when considering reproducibility. For example, reporting results on a shareable dataset, even with limitations such as size or scope, could be helpful.

Could WMS be used in NLP?

NLP tools could easily be adapted to be used through WMS. The main requirement for using WMS is a clear definition of inputs and outputs. A large number of NLP tools already fulfill this requirement; clinical NLP shared tasks (eg, offered at n2c2, CLEF, SemEval) have encouraged the adoption of well-defined formats implemented in the shared datasets.[53] The availability of the tools as a command line executable (ie, not as graphical user interfaces) also simplifies the integration process. For example, QuickUMLS[54] can easily be integrated into the WMS Nextflow[16] (eg, https://github.com/equipe22/QuickUMLS_Nextflow).

Moreover, the combination of single tools in WMS is simple. The workflows are modeled as graphs, with nodes corresponding to tools and edges corresponding to data streams between tools.

Lessons learned from bioinformatics: modularity and workflows contribute to reproducibility

Clinical NLP and bioinformatics share important characteristics. In both cases, the raw data can be viewed as a sequence of words, tools are dedicated to the annotation of the sequence, often relying on offsets (start and end indices) and external resources (reference sequences in bioinformatics, standard terminologies in clinical NLP).

Using workflow management systems to support reproducible and repeatable analysis

Faced with 2 bottlenecks, namely (i) the increase in data complexity and volume, and (ii) the growing demand for reproducible science, the field of bioinformatics has applied workflow management systems. WMS were designed to ensure the proper behavior of tools developed by various groups and teams across the world. Furthermore, tools not necessarily designed to be used together could be unified within WMS. Today, Snakemake, Galaxy, and Nextflow are adopted by the bioinformatics community. WMS are associated with 2 communities. The developer community uses all WMS with a preference for Snakemake and Nextflow. They use preferentially Galaxy to share their tools with the nondeveloper community which mainly consists of Galaxy users. Galaxy provides an out-of-the-box solution to ease the shareability and usability of tools for nonspecialists, as it provides a user interface for pipeline construction. Further adoption leveraging the full power of WMS (ie, scalability, traceability, etc) would probably be beneficial in NLP.

Enhance modularity and shareability with containers

The variety of tools available in the bioinformatics community has led to the adoption of containerization technologies (eg, Docker or Singularity). Containers have become standard ways of distributing tools. Recently, repositories of containers such as Biocontainers[55] have emerged. While some NLP tools (eg, Metamap[56]) are distributed as containers, there is still an opportunity for improvement. To the best of our knowledge, there are no public container repositories in NLP.

Implementation and adoption of standard exchange format

Bioinformatics has adopted standard input and output files format. For example, DNA sequences are stored as FastQ files, aligned sequences as BAM (or SAM) files, and identified genomics variants (clinically useful information) as VCF files. Each of the formats corresponds to a specific set of processes. Ontologies of formats have been developed,[57] allowing a formal description of input, output, and tools. Despite the emergence of standard exchange formats in NLP (such as BioC[58] or CoNLL[59]) their adoption remains low.

Using provenance for traceability of processes and results

The bioinformatics community has been using the concept of provenance over the last decade. For example, provenance is managed within the Galaxy History.[18,25] Additionally, the provenance ontology PROV-O has been used in the field of NLP (eg, ProvCaRe[23]) A better integration of PROV-O could enhance traceability in NLP frameworks.

Public repository of curated workflows

While WMS ensure a basic level of reproducibility, the existence of a repository enabling the exchange and improvement of curated workflow can be helpful in that regard. Recently, the community of Nextflow users released nf-core,[60] a shared repository of curated workflows. To the best of our knowledge, a public repository of curated workflows does not exist in the field of NLP despite the presence of curated workflow in NLP frameworks such as CLAMP or cTakes.

From replicability to reusability

As a motivation for this study, we outlined the importance of achieving replicability (reiterating experiments in identical settings to yield the same results) to pave the way to reusability (conducting similar experiments with selected variations in methods and/or datasets to yield consistent conclusions). The core of clinical NLP research investigates methods and the right combination of NLP methods and clinical data to achieve clinical outcomes. From the perspective of replication, methods are not investigated as such, since the goal is to apply existing tools, models, and methods on corpora without modification (ie, without retraining machine learning models, recreating deep semantic representations, or information extraction rules). In this context, traceability offers the means to facilitate the comparison to original experiments and identify discrepancies. Traceability of performance at each step in a workflow helps identify the cause for overall variation in a workflow. Moving on to reproducibility and reusability, changes to corpora or methods can be introduced. The importance of traceability is therefore heightened to keep track of the new experiments specifics as well as to better control for workflow variations. In this context, parameters specific to machine learning methods need to be tracked, such as corpus partitions, random seeds, learning rate, and size of hidden layers. Likewise, resources, such as language models or terminologies used by NLP methods, can be customized or retrained, which needs to be recorded. Additionally, performance obtained on a given dataset with a specific set of parameters should also be recorded—ideally, using a shareable dataset for this purpose. Deploying NLP pipelines on locally customized data (R35) and tasks (R36) while achieving comparable outcomes presents additional challenges[12] investigated in the bioNLP community as domain adaptation.

CONCLUSION

In summary, we identified 40 reproducibility recommendations based on the review of 8 articles from heterogeneous research fields (bioinformatics, medical informatics, and NLP) and our daily experience of the deployment and use of NLP processes in a clinical setting. We assigned a level of application to each feature to support the development of reproducible tools and pipelines. Finally, we investigate the adherence of 7 frameworks to these features. We noticed that the NLP framework implemented within Galaxy WMS, implemented more than 50% of the features. NLP frameworks could take advantage of lessons learned from other fields (and especially bioinformatics) to improve reproducibility for NLP systems in clinical settings. More precisely, specific features could be transferred to clinical NLP, such as public repositories of curated workflows, enhanced modularity, and shareability with containers or provenance information for traceability of processes and results. We believe that reproducibility is a necessary—although not sufficient—intermediate step towards the reuse of NLP tools, including modern neural methods. Versioning and reproducibility (including distribution of sample open data) are actionable steps that complement FAIR principles to empirically verify the validity of tools used in a new environment.

FUNDING

WD and DB were as supported in part by the ANR PractikPharma grant (ANR-15-CE23-0028) from the French Agence Nationale de la Recherche. BR was supported in part by the SIRIC CARPEM research program.

AUTHOR CONTRIBUTIONS

WD, ANL, and BR initiated the work. WD collected the data, performed the analysis, and wrote the first draft. All authors provided scientific inputs and contributed to the manuscript. WD, ANL, and BR heavily revised the manuscript and responded to reviews. All the authors approved the final version of the manuscript.

SUPPLEMENTARY MATERIAL

Supplementary material is available at the *Journal of the American Medical Informatics Association* online.

CONFLICT OF INTEREST STATEMENT

None declared.

REFERENCES

1. Baker M. 1,500 scientists lift the lid on reproducibility. *Nature News* 2016; 533 (7604): 452–4.
2. Cohen-Boulakia S, Belhajjame K, Collin O, *et al*. Scientific workflows for computational reproducibility in the life sciences: status, challenges and opportunities. *Future Gen Comput Syst* 2017; 75: 284–98.
3. Wilkinson MD, Dumontier M, Aalbersberg IJ, *et al*. The FAIR Guiding Principles for scientific data management and stewardship. *Sci Data* 2016; 3 (1): 1–9.
4. Collberg C, Proebsting TA. Repeatability in computer systems research. *Commun ACM* 2016; 59 (3): 62–9.
5. Benureau FCY, Rougier NP. Re-run, repeat, reproduce, reuse, replicate: transforming code into scientific contributions. *Front Neuroinform* 2018; 11: 1–8.
6. Marx V. When computational pipelines go 'clank.' *Nat Methods* 2020; 17 (7): 659–62.
7. Pedersen T. Empiricism is not a matter of faith. *Comput Linguistics* 2008; 34 (3): 465–70.
8. Fokkens A, Erp MV, Postma M, *et al*. Offspring from Reproduction Problems: What Replication Failure Teaches Us. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2013: 1691–701.
9. Cohen KB, Xia J, Zweigenbaum P, *et al*. Three dimensions of reproducibility in natural language processing. *LREC Int Conf Lang Resour Eval* 2018; 2018: 156–65.
10. Bakken S. The journey to transparency, reproducibility, and replicability. *J Am Med Inform Assoc* 2019; 26 (3): 185–7.
11. Velupillai S, Suominen H, Liakata M, *et al*. Using clinical natural language processing for health outcomes research: overview and actionable suggestions for future advances. *J Biomed Inform* 2018; 88: 11–9.
12. Carrell DS, Schoen RE, Leffler DA, *et al*. Challenges in adapting existing clinical natural language processing systems to multiple, diverse health care settings. *J Am Med Inform Assoc* 2017; 24 (5): 986–91.
13. Sandve GK, Nekrutenko A, Taylor J, *et al*. Ten simple rules for reproducible computational research. *PLoS Comput Biol* 2013; 9 (10): e1003285.
14. Noble WS. A quick guide to organizing computational biology projects. *PLoS Comput Biol* 2009; 5 (7): e1000424.
15. Lee BD. Ten simple rules for documenting scientific software. *PLoS Comput Biol* 2018; 14 (12): e1006561.

16. Di Tommaso P, Chatzou M, Floden E, P, *et al.* Nextflow: enables reproducible computational workflows. Nat biotechnol 35 (4): 316–9.
17. Köster J, Rahmann S. Snakemake–a scalable bioinformatics workflow engine. *Bioinformatics* 2012; 28 (19): 2520–2.
18. Afgan E, Baker D, van den Beek M, *et al.* The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. *Nucleic Acids Res* 2016; 44 (W1): W3–10.
19. Docker, Inc. Docker - Build, Ship, and Run Any App, Anywhere. https://www.docker.com/ Accessed September 30, 2020
20. Linux Containers. https://linuxcontainers.org/Accessed September 30, 2019
21. Kurtzer GM, Sochat V, Bauer MW. Singularity: scientific containers for mobility of compute. *Plos One* 2017; 12 (5): e0177459.
22. Bánáti A, Kacsuk P, Kozlovszky M. Four level provenance support to achieve portable reproducibility of scientific workflows. In: proceedings of the 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO 2015-Proceedings). 2015: 241–4. doi: 10.1109/MIPRO.2015.7160272.
23. Valdez J, Kim M, Rueschman M, *et al.* ProvCaRe semantic provenance knowledgebase: evaluating scientific reproducibility of research studies. *AMIA Annu Symp Proc* 2017; 2017: 1705–14.
24. Khan FZ, Soiland-Reyes S, Sinnott RO, *et al.* Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv. *Gigascience* 2019; 8 (11): 1–27. doi: 10.1093/gigascience/giz095.
25. Gaignard A, Belhajjame K, Skaf-Molli H. SHARP: Harmonizing Galaxy and Taverna workflow provenance. In: proceedings of the *SeWeBMeDA 2017: Semantic Web solutions for large-scale BioMedical Data Analtics*. Portoroz, Slovenia: 2017. https://hal.archives-ouvertes.fr/hal-01768401Accessed October 1, 2019
26. PROV-O: The PROV Ontology (Manchester eScholar—The University of Manchester). https://www.escholar.manchester.ac.uk/jrul/item/? pid=uk-ac-man-scw : 231770Accessed September 30, 2019
27. Ide N, Suderman K, Pustejovsky J, *et al.* The Language Application Grid and Galaxy. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA); 2016: 457–62.
28. Labropoulou P, Galanis D, Lempesis A, *et al.* OpenMinTeD: A Platform Facilitating Text Mining of Scholarly Content. In: *WOSP 2018 Workshop Proceedings*. Luxemburg: European Language Resources Association (ELRA) 2018. http://lrec-conf.org/workshops/lrec2018/W24/pdf/13_W24.pdfAccessed October 10, 2019
29. Tricco AC, Lillie E, Zarin W, *et al.* PRISMA extension for scoping reviews (PRISMA-ScR): checklist and explanation. *Ann Intern Med* 2018; 169 (7): 467–73.
30. McIntosh LD, Juehne A, Vitale CRH, *et al.* Repeat: a framework to assess empirical reproducibility in biomedical research. *BMC Med Res Methodol* 2017; 17 (1): 143.
31. Taschuk M, Wilson G. Ten simple rules for making research software more robust. *PLOS Comput Biol* 2017; 13 (4): e1005412.
32. Savova GK, Masanz JJ, Ogren PV, *et al.* Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation, and applications. *J Am Med Inform Assoc* 2010; 17 (5): 507–13.
33. Soysal E, Wang J, Jiang M, *et al.* CLAMP—a toolkit for efficiently building customized clinical natural language processing pipelines. *J Am Med Inform Assoc* 2018; 25 (3): 331–6.
34. Cunningham H, Tablan V, Roberts A, *et al.* Getting more out of biomedical documents with GATE's full lifecycle open source text analytics. *PLOS Comput Biol* 2013; 9 (2): e1002854.
35. Neumann M, King D, Beltagy I, *et al.* ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In: *BioNLP@ACL*. 2019. doi: 10.18653/v1/W19-5034
36. Perovšek M, Kranjc J, Erjavec T, *et al.* TextFlows: A visual programming platform for text mining and natural language processing. *Sci Comput Programming* 2016; 121: 128–52.
37. Cunningham H. GATE, a General Architecture For Text Engineering. *Comput Hum* 2002; 36 (2): 223–54.
38. Team CD. CLAMP | Natural Language Processing (NLP) Software. https://clamp.uth.edu/manual.php#output_visualizationAccessed 22 June, 2020
39. Stenetorp P, Pyysalo S, Topić G, *et al.* brat: a Web-based Tool for NLP-Assisted Text Annotation. In: *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Avignon, France: Association for Computational Linguistics 2012. 102–7.https://www.aclweb.org/anthology/E12-2021Accessed Jun 22, 2020
40. Carpenter B. LingPipe for 99.99% Recall of Gene Mentions. In: *Proceedings of the Second BioCreative Challenge Evaluation Workshop*. BioCreative. 2007; 23: 307–9.
41. Apache OpenNLP.Text Annotation with OpenNLP and UIMA. https://opennlp.apache.org/ Accessed Jun 22, 2020. G
42. spaCy. Industrial-strength natural language processing in Python. https://spacy.io/Accessed 28 Aug 28, 2020
43. Kranjc J, Podpečan V, Lavrač N. ClowdFlows: a cloud based scientific workflow platform. In: Flach PA, De Bie T, Cristianini N, eds. *Machine Learning and Knowledge Discovery in Databases*. Berlin: Springer; 2012: 816–9.
44. Bird S. NLTK: the natural language toolkit. In: *Proceedings of the COLING/ACL on Interactive presentation sessions*. Sydney, Australia: Association for Computational Linguistics; 2006: 69–72.
45. Pedregosa F, Varoquaux G, Gramfort A, *et al.* Scikit-learn: Machine Learning in Python. *Machine Learning in Python. https://scikit-learn.org/stable/index.html* Accessed Jun 22, 2020.
46. Manning C, Surdeanu M, Bauer J, *et al.* The Stanford CoreNLP natural language processing toolkit. In: *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Baltimore, MD: Association for Computational Linguistics; 2014: 55–60.
47. Apache OpenNLP. https://opennlp.apache.org/ Accessed Jun 22, 2020.
48. Ferrucci D, Lally A. UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Nat Lang Eng* 2004; 10 (3-4): 327–48.
49. Johnson AEW, Pollard TJ, Shen L, *et al.* MIMIC-III, a freely accessible critical care database. *Sci Data* 2016; 3 (1): 160035.
50. Grabar N, Claveau V, Dalloux C. CAS: French corpus with clinical cases. In: *Proceedings of the Ninth International Workshop on Health Text Mining and Information Analysis*. Brussels, Belgium: Association for Computational Linguistics; 2018: 122–8. doi: 10.18653/v1/W18-5614.
51. Névéol A, Grouin C, Leixa J, *et al.* The Quaero French medical corpus: A ressource for medical entity recognition and normalization. In: *In Proceedings of Bio text-mining workshop (BioTextM), Reykjavik*. 2014.
52. Sarker A, Belousov M, Friedrichs J, *et al.* Data and systems for medication-related text classification and concept normalization from Twitter: insights from the Social Media Mining for Health (SMM4H) 2017 shared task. *J Am Med Inform Assoc* 2018; 25 (10): 1274–83.
53. Chapman WW, Nadkarni PM, Hirschman L, *et al.* Overcoming barriers to NLP for clinical text: the role of shared tasks and the need for additional creative solutions. *J Am Med Inform Assoc* 2011; 18 (5): 540–3.
54. Soldaini L, Goharian N. QuickUMLS: a fast, unsupervised approach for medical concept extraction. In: Proceedings in Medical Information Retrieval (MedIR) Workshop, *sigir*. 2016: 1–4.
55. da Veiga Leprevost F, Grüning BA, Alves Aflitos S, *et al.* BioContainers: an open-source and community-driven framework for software standardization. *Bioinformatics* 2017; 33 (16): 2580–2.
56. Aronson AR. Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *Proc AMIA Symp* 2001; 17–21.
57. Ison J, Kalaš M, Jonassen I, *et al.* EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats. *Bioinformatics* 2013; 29 (10): 1325–32.
58. Comeau DC, Islamaj Dogan R, Ciccarese P, *et al.* BioC: a minimalist approach to interoperability for biomedical text processing. *Database (Oxford)* 2013; 2013 (0): bat064.
59. Buchholz S, Marsi E. CoNLL-X shared task on multilingual dependency parsing In: *Proceedings of the Tenth Conference on Computational Natu-*

ral Language Learning (CoNLL-X). New York City:Association for Computational Linguistics; 2006. 149–64.https://www.aclweb.org/anthology/W06-2920Accessed June 22, 2020

60. Ewels PA, Peltzer A, Fillinger S, *et al*. The nf-core framework for community-curated bioinformatics pipelines. *Nat Biotechnol* 2020; 38 (3): 276–3.