A peer-reviewed version of this preprint was published in PeerJ on 31 October 2018.

View the peer-reviewed version (peerj.com/articles/5853), which is the preferred citable publication unless you specifically need to cite this preprint.

Li J, Cui B, Dai Y, Bai L, Huang J. 2018. BioInstaller: a comprehensive R package to construct interactive and reproducible biological data analysis applications based on the R platform. PeerJ 6:e5853 https://doi.org/10.7717/peerj.5853

BioInstaller: a comprehensive R package to construct interactive and reproducible biological data analysis applications based on the R platform

Jianfeng Li [1], **Bowen Cui** [1], **Yuting Dai** [1,2], **Ling Bai** [1], **Jinyan Huang** [Corresp. 1]

[1] State Key Laboratory of Medical Genomics, Shanghai Institute of Hematology, National Research Center for Translational Medicine, Rui-Jin Hospital, Shanghai Jiao Tong University School of Medicine, Shanghai Jiao Tong University, Shanghai, China
[2] School of Life Sciences and Biotechnology, Shanghai Jiao Tong University, Shanghai, China

Corresponding Author: Jinyan Huang
Email address: jinyan@shsmu.edu.cn

The increase in bioinformatics resources such as tools/scripts and databases poses a great challenge for users seeking to construct interactive and reproducible biological data analysis applications. Here, we propose an open-source, comprehensive, flexible R package named BioInstaller that consists of the R functions, Shiny application, the HTTP representational state transfer (REST) application programming interfaces (APIs), and a docker image. BioInstaller can be used to collect, manage and share various types of bioinformatics resources and perform interactive and reproducible data analyses based on the extendible Shiny application with Tom's Obvious, Minimal Language (TOML) and SQLite format databases. The source code of BioInstaller is freely available at our lab website, http://bioinfo.rjh.com.cn/labs/jhuang/tools/bioinstaller, the popular package host GitHub, https://github.com/JhuangLab/BioInstaller, and the Comprehensive R Archive Network (CRAN), https://CRAN.R-project.org/package=BioInstaller. In addition, a docker image can be downloaded from DockerHub (https://hub.docker.com/r/bioinstaller/bioinstaller).

BioInstaller: a comprehensive R package to construct interactive and reproducible biological data analysis applications based on the R platform

Jianfeng Li[1,2,*], Bowen Cui[1,*], Yuting Dai[2], Ling Bai[1], and Jinyan Huang[1]

[1] State Key Laboratory of Medical Genomics, Shanghai Institute of Hematology, National Research Center for Translational Medicine, Rui-Jin Hospital, Shanghai Jiao Tong University School of Medicine, Shanghai Jiao Tong University, Shanghai, China

[2] School of Life Sciences and Biotechnology, Shanghai Jiao Tong University, Shanghai, China

[*] These authors contributed equally to this work.

Corresponding author: Jinyan Huang, jinyan@shsmu.edu.cn

ABSTRACT

The increase in bioinformatics resources such as tools/scripts and databases poses a great challenge for users seeking to construct interactive and reproducible biological data analysis applications. Here, we propose an open-source, comprehensive, flexible R package named BioInstaller that consists of the R functions, Shiny application, the HTTP representational state transfer (REST) application programming interfaces (APIs), and a docker image. BioInstaller can be used to collect, manage and share various types of bioinformatics resources and perform interactive and reproducible data analyses based on the extendible Shiny application with Tom's Obvious, Minimal Language (TOML) and SQLite format databases. The source code of BioInstaller is freely available at our lab website, http://bioinfo.rjh.com.cn/labs/jhuang/tools/bioinstaller, the popular package host GitHub, https://github.com/JhuangLab/BioInstaller, and the Comprehensive R Archive Network (CRAN), https://CRAN.R-project.org/package=BioInstaller. In addition, a docker image can be downloaded from DockerHub (https://hub.docker.com/r/bioinstaller/bioinstaller).

INTRODUCTION

With the rapid development of new bioscience technology, particularly next-generation sequencing (NGS), volumes of "omics" data have been generated, such as the 1000 Genomes Project, The Cancer Genome Atlas (TCGA), and Genotype-Tissue Expression (GTEx) (Abecasis et al. 2012; Cancer Genome Atlas Research et al. 2013; Consortium 2013; Sanchez-Vega et al.

2018). The bioinformatics tools and databases required for the downstream data analysis are also increasing at a phenomenal rate. R language, as the most popular programming language for statistics, biological data analysis, and big data, has enabled diverse and free R packages (>14000) for different types of applications, such as high throughput sequencing data analysis (e.g. Bioconductor) (Gentleman et al. 2004) and the development of web applications (e.g. Shiny framework) (Chang et al. 2015). With the development of web technologies and the release of the R web developmental framework Shiny, the number of interfaces available to R users has increased. However, due to the lack of high-performance and open-source cloud platforms based on R (e.g., Galaxy for Python users) (Afgan et al. 2016), it is still difficult for R users, especially those without web development skills, to construct interactive and reproducible biological data analysis applications supporting the upload and management of files, long-time computation, task submission, tracking of output files, exception handling, logging, export of plots and tables, and extendible plugin systems.

Another common problem usually faced by R and other programming platform users (e.g., the team of Galaxy) (Afgan et al. 2016) is how to acquire and share certain bioinformatics resources quickly and accurately. Numerous bioinformatics tools (e.g., primer design, sequence alignment, variant calling and annotation) or scripts (e.g., data format conversion, text processing) are scattered around world web hosts. Biomedical databases are facing the same situation. For example, genome sequences (e.g., hg19/hg38 for human, mm9 and mm10 for mouse) are mainly deposited in the UCSC Genome Browser and National Center for Biotechnology Information (NCBI, https://www.ncbi.nlm.nih.gov/) (Tyner et al. 2017). The best-known gene and transcript annotation resources are provided by GENCODE and the RefSeq database (Derrien et al. 2012; O'Leary et al. 2016). Genetic variants annotation databases, mainly cancer and Mendelian disorder related, are hosted by the original projects, e.g. TCGA, and various down-stream tools, e.g., ANNOVAR, Variant Effect Predictor (VEP), Oncotator (McLaren et al. 2016; Ramos et al. 2015; Wang et al. 2010). Bioconductor is a popular bioinformatics R community for sharing genetic variants and other types of bioinformatics annotation databases via R package (Gentleman et al. 2004), but it is difficult for users to share many types of tools/scripts and databases if they do not have the capability of packing their own tools/scripts and databases. In most cases, these resources are isolated and can only be accessed via a command line tool such as rsync (https://rsync.samba.org/) or wget (http://www.gnu.org/software/wget/), to request the

corresponding Uniform Resource Locators (URLs). Software distribution tools that do not demand root privileges, such as conda (https://github.com/conda/conda) and spack (Gamblin et al. 2015), have greatly improved the acquisition of bioinformatics software. However, considering the huge growth of tools/scripts and databases required for bioinformatics data analysis, the resources supported by these software distribution tools are far from sufficient. Users also need more experience to use these different package management tools under command line environment.

Here, we present an open-source, comprehensive, flexible bioinformatics platform named BioInstaller that can be used to collect, manage and share various types of bioinformatics resources and to perform interactive and reproducible data analyses. By utilizing a simplified and standard TOML format configuration file with extra parse functions, the developers and users can freely and unreservedly share their public or internal bioinformatics tools/scripts and databases online on the GitHub repository or other hosts. In addition, users can easily obtain access to pooled bioinformatics resources via the diverse interfaces of BioInstaller, which includes R functions, the Shiny application (Chang et al. 2015) and HTTP representational state transfer (REST) application programming interfaces (APIs) that are rarely adopted in other similar tools. As a practical demonstration, we collected 157 tools/scripts and 110 databases specifically related to genetic variants annotation using the BioInstaller-defined configuration files. Notably, we developed a Shiny application to support functions including system monitoring, the logging system, file management, the queue system, and so on. This application can easily be reused in other Shiny applications. We expect the BioInstaller package and the practices in this work to reduce the difficulty of constructing the interactive and reproducible biological data analysis applications for R users, and to further improve the interactivity and reproducibility of bioinformatics data analysis.

MATERIAL AND METHODS

Design and development of BioInstaller

BioInstaller was designed as an interactive R package to collect, manage and share various types of bioinformatics resources and perform interactive and reproducible data analyses. BioInstaller contains the R functions and the Shiny application (Chang et al. 2015) and REST APIs (**Figure 1**). Both R and other programming platform users can utilize the functions of BioInstaller, such as by downloading bioinformatics tools/scripts and databases and performing statistical analysis and visualization. The R and Shiny interfaces of BioInstaller were mainly developed in R language

and utilize the HTML/CSS and JavaScript languages. To run an instance of BioInstaller, the R program and extra dependent R packages are required. Travis CI (https://www.travis-ci.org/) was used to automatically test the R functions on Linux and MAC OSX platforms. Periodically, the tested and updated BioInstaller package is submitted to CRAN with an increased version number, e.g., from v3.3.3 to v3.3.4. Both the open and restricted bioinformatics resources can be integrated using the TOML format configuration file. The configuration files can also be used in other programming language platforms to access desired masteries by using a unique item name, such as 'bwa', 'gatk', 'annovar', 'db_annovar_1000g', 'db_annovar_gtex', etc. A hash value was generated using the item name and version for the unique ids of tools/scripts and databases. An autogenerated docker image containing all required R packages and the backend web service of BioInstaller have been deposited at the DockerHub (https://hub.docker.com/r/bioinstaller/bioinstaller).

GitHub API and custom values/functions for querying of version

The querying of versions of bioinformatics tools/scripts and databases of a GitHub or non-GitHub project is the basic function of BioInstaller. For GitHub projects items, the GitHub APIs were used to access the projects version information, such as release, tags, and branches. All released versions will be used as the available versions and returned to BioInstaller (**Figure S1a**). However, the situation becomes more complicated if the resources have not been published on GitHub. Here, we propose two types of methods of parsing item versions. Method I: If the released versions are fixed, users can write it in the 'version_available' field in the configuration file. Method II: Utilizing the R packages rvest (https://CRAN.R-project.org/package=rvest) and RCurl (https://CRAN.R-project.org/package= RCurl) (Wu et al. 2016), we established an R functions pool to dynamically query the version of items from the original release website (**Dataset S1**). The demo function to query the latest version of GMAP is shown in **Figure S1b**. This is useful for automating a pipeline to build the precompiled binary version.

Mirror resource for an invalid link

Network transferring is a common problem in bioinformatics data analysis. A mirror resource is one option to partially resolve these problems, including an invalid link and network blocking. BioInstaller allows users to set any numbers of mirror URLs for their tools/scripts and databases

to avoid the possible problems caused by network transmission. As shown in **Figure S1c,** the mirror URLs of Miniconda (https://conda.io/miniconda.html) are separately provided by the official and our hosts. Notably, established mirror URLs of bioinformatics resources can be used in the spack (Gamblin et al. 2015) and other similar tools to build the cache files.

TOML format configuration files

Massive bioinformatics tools/scripts and databases have been integrated into BioInstaller. Tom's Obvious, Minimal Language (TOML) is a popular and human-readable configuration formats supporting comments. We uses standard TOML format configuration file to store required information of the included bioinformatics tools/scripts and databases. These configuration files can be reused in other bioinformatics software packages or data-analysis pipelines via online accession or as a file copy. We have provided six directories to store different types of TOML files including 'github', 'nongithub', 'database', 'web', 'docker', and 'shiny'. Due to the broad compatibility of BioInstaller, any resource published on docker, GitHub, Zenodo (https://zenodo.org/) or other platforms can be supported.

Implementation of the Shiny application

To increase the convenience of BioInstaller for nonprogramming users, a user-friendly web application was developed based on Shiny (Chang et al. 2015). The user interface (UI) of BioInstaller was constructed using the R package shinydashboard (https://cran.r-project.org/package=shinydashboard) and Shiny (Chang et al. 2015). Output tables were generated by the R package DT (https://CRAN.R-project.org/package=DT) and wrapped JavaScript library DataTables (https://datatables.net/). Charts were mainly generated by published R packages and in-house scripts or R packages that all support interactive update and export of PDF, SVG, and PNG format plots. The tab items of the BioInstaller Shiny application at the left side of the navigation bar can be used to switch among all available modules, including 'Introduction', 'Dashboard', 'Upload', 'File Viewer', 'Pipeline', 'Instant', 'Installer', and 'Setting'. The detail usage guidelines are provided on our host (http://bioinfo.rjh.com.cn/labs/jhuang/tools/BioInstaller/), and R users can also use the browser vignettes functions in R to access these documents.

RESULTS

Overview and practices of BioInstaller's functionalities

A comprehensive R package was developed that could be used to quickly construct interactive and reproducible biological data analysis applications based on the R platform (**Figure 2**). The functionalities (**Table 1, Dataset S2**) of BioInstaller were divided into six parts based on whether users use BioInstaller or not: 1) deployment of resources, 2) collection of resources, 3) sharing of resources, 4) construction of pipelines, 5) construction of Shiny applications, and 6) reproducible data analysis. An example of a real project (annovarR, https://github.com/JhuangLab/annovarR, under development) is shown in **Figure 2** to illustrate the full workflow for BioInstaller utilization**,** which was designed to integrate various genetic variant annotation and visualization tools, including public command line tools, R packages and custom annotation and visualization functions. Using the code library, predefined TOML files (database resources and plugins), and the docker file of BioInstaller, we could easily customize the BioInstaller-established Shiny application to work on the genetic variants annotation tasks. If BioInstaller is not used, we need to develop the UI and server code of the Shiny application for a large number of universal functions, such as the file management system, background task submission and queue management, and tracking of the output log and files. The docker image of BioInstaller is also out-of-the-box and could be modified and applied to our own works. Based on the integrated installer (e.g., conda, spack, and BioInstaller) and simplified TOML files of BioInstaller, users can collect, share, and deploy genetic variant annotation databases and tools with one-stop service. As a real practice of BioInstaller, we collected and shared more than 157 tools/scripts and 110 databases (**Table 2, S1, S2**) in the configuration pool of BioInstaller, including genetic variant annotation databases and tools; the meta information is freely available and hosted on the public GitHub website (https://github.com/JhuangLab/BioInstaller/tree/master/inst/extdata/config). The raw files are stored on the original websites (e.g., https://github.com, https://sourceforge.net/, http://annovar.openbioinformatics.org/, etc.) and our host.

Comparison of BioInstaller with existing tools for the collection and sharing of bioinformatics resources

To better understand the advance provided by BioInstaller in terms of the collection and sharing of bioinformatics resources, we further compared BioInstaller with several existing tools, including Omictools (Henry et al. 2014) and Datasets2Tools (Torre et al. 2018) (**Table 1, 2**), the

two most comprehensive meta databases focused on bioinformatics tools. All provide a web forum to update the meta database of bioinformatics resources. However, BioInstaller offers an off-line way to develop the users' own meta databases via an unlimited configuration file pool (TOML and SQLite format) that is easy to carry and share and is independent of programming knowledge. In addition, the developed R functions and Shiny application can be used to query and download the linked or isolated file databases, such as appendix data from papers, annotation databases for genetic variants, genome sequences, etc. In most cases, it is suitable to tightly combine the meta database with the file database. Therefore, we designed and shared an upload module in the Shiny application to set the meta information for all files, and users can add the description, genome version, custom file types, and other customizable fields. Both Omictools and Dataset2Tools only include the items in their databases and do not integrate external resources. BioInstaller not only can be used to collect users own resources, but also can be used to integrate external resources.

Summary of supported bioinformatics tools/scripts and databases

For now, 157 tools/scripts and 110 databases are natively supported in BioInstaller (**Figure 1**, **Table 2, S1, S2**). First, we covered the most commonly used tools in each bioinformatics analysis process, including data quality control (n=17), alignment and assembly (n=27), variant detection (n=32) or annotation (n=12), high-throughput sequence (HTS) manipulation (n=17) and visualization libraries (n=11) (**Table 1, S1**), etc. Second, BioInstaller also provides abundant databases for annotating data or satisfying software dependencies. With BioInstaller, users can easily download UCSC sequence and annotation data (n=4995) (**Dataset S3**), blast databases (n=29) (**Table S2**), allele frequency databases (n=17), variant effect prediction databases (n=29), and disease-related (n=13), drug-related (n=4), noncoding region-related databases (n=15) (**Table 2, S2**), among others. Notably, we collected and constructed 20 genetic variant annotation databases, which can be directly used in other variants annotation tools, including ANNOVAR (Wang et al. 2010), vcfanno (Pedersen et al. 2016), and annovarR (https://github.com/JhuangLab/annovarR).

BioInstaller has been released on CRAN for one and a half years and has accumulated a certain number of users, with a total of 19,912 downloads from CRAN (2018.8.3). In the recent release (v0.3.5), we provided the Shiny application and significantly expanded the supported tools/scripts and databases. The number of supported tools/scripts and databases is still increasing and is being

applied to other related projects, such as the integrated genetic variants annotation tool annovarR (https://github.com/JhuangLab/annovarR).

Examples of BioInstaller R functions

We have demonstrated the basic structure, functions, and web service of BioInstaller. The full help document is available at http://bioinfo.rjh.com.cn/labs/jhuang/tools/BioInstaller/articles/. Because most of the Shiny application UIs are wrapped with R functions, we use several use examples to illustrate the R functions of BioInstaller.

Example #1: Install packed or unpacked bioinformatics tools. We use the Ion Torrent Variant Caller (TVC) (Zook et al. 2014) and svaba (Wala et al. 2018) to show how to install or download the bioinformatics tools or scripts that are not supported by other package management tools.

```
> library(BioInstaller) # Library the R package
> set.biosoftwares.db("~/.BioInstaller/info.yaml") # Store the installation information
> install.bioinfo(show.all.names = TRUE) # Get all items name supported by BioInstaller
> install.bioinfo(name = "tvc", show.all.versions = TRUE) # Get all available versions of tvc
> install.bioinfo(name = "svaba", show.all.versions = TRUE) # Get all available versions of
svaba
> install.bioinfo(name='tvc', download.dir = "/path/tool/tvc") # One-click install the tvc
> install.bioinfo(name='svaba, download.dir = "/path/tool/svaba") # One-click install the svaba
> show.installed() # Get all installed tools
> get.info("svaba") # Get the svaba installation information, such as update time and version
```

Example #2: Download genetic variants annotation databases. Genetic variants annotation is a common and high-demand task for most biomedicine research, especially for examining the correlations between phenotype and molecular features, such as germline and somatic mutations. The followed example describes how to download the genetic variants annotation databases dbSNP, CIViC, DisGeNET, and CancerHotspot (Chang et al. 2016; Griffith et al. 2017; Piñero et al. 2017).

```
> install.bioinfo("db_annovar_avsnp", extra.list = list(buildver = "hg19"), download.dir =
"/path/db/") # install the latest dbSNP from ANNOVAR website
```

> crawl.all.versions("db_annovar_avsnp") # Download all dbSNP to current directory

> install.bioinfo("db_civic", download.dir = "/tmp/db") # Download the nightly version of CIViC database

> install.bioinfo("db_disgenet", download.dir = "/tmp/db") # Download the DisGeNET database

> install.bioinfo("db_cancer_hotspots", download.dir = "/tmp/db") # Download the DisGeNET databaseß

Example #3: Download an annotation database based on the supplementary files of published papers. The followed example is an epigenetic genes classification (e.g., reader, writer, eraser) database only available in the papers supplementary file (Huether et al. 2014).

> install.bioinfo("db_annovar_epi_genes", extra.list = list(buildver = "hg19"), download.dir = "/path/db/") # install the epigenetic genes database from our website

User-interfaces and functions of the Shiny application

Introduction module. Utilizing the Shiny function 'includeMarkdown', we generated the 'Introduction' module page from Markdown, a lightweight markup language, format document (**Figure S2a**).

Dashboard module. The 'Dashboard' module includes the system monitors, such as hardware (Disk and memory), queue tasks, task log, installed R packages, Python packages, conda environments, and the other information of the operating environments (**Figure S2b, c, S3**). The monitored data stream is automatically updated once every 10 seconds (**Figure S2b**). A demo table output in the dashboard lists all files in the environment variable 'PATH', where users can use the selector at the lower left quarter to customize the row numbers (5, 10, 25, 50, and all) (**Figure S2c**). All output tables in BioInstaller can be easily exported to CSV, XLS, PDF files or directly copied to the clipboard. Monitor plugins related to the information of the R system (**Figure S3a, b, c**), BioInstaller (**Figure S3d**), conda (**Figure S3e**) and spack **(Figure S3f)** are integrated in this work, which can reduce user input of extra command line commands and facilitate sharing with others.

Upload module. The 'Upload' module is used to upload files to the BioInstaller Shiny web platform. Optional fields, such as file type, genome version, and description, can be stored in the SQLite format database with the uploaded files path and the files md5 value (**Figure S2d**). When

uploading a file, users need to click the 'Save' button to confirm the upload behavior and update the database (**Figure S2e**). Before the confirmation click, users can preview the file and make a final decision (**Figure S2f**). Files with sizes ranging from 0.25 GB to 8 GB were tested on the Shiny application (**Table S3**). For files larger than 10 GB, we recommend using the rsync or FTP service to transfer files and then adding the corresponding description and records in TOML or SQLite databases.

File viewer module. The 'File viewer' module is used to manage all uploaded files in the BioInstaller Shiny application that supports view, delete and download, and all files can be used in the other plugins of the BioInstaller Shiny application, mainly in the 'Pipeline' and 'Instant' modules (**Figure 3a, b**).

Pipeline module. The 'Pipeline' module is used to integrate complicated bioinformatics analysis workflows or other small scripts. An in-house interpreter R function was used to parse the plugin configuration files to generate the Shiny UI and server functions. A small script creating a data analysis directory structure was used as the demo for 'Pipeline' (**Dataset S4**). Users can input the project name and the parent directory to create a series of predefined directories. The R commands used in this task are editable at the bottom of the box (**Figure 3c**). After users click the 'Submit' button, BioInstaller will generate a random character as the submitted task key. Users can use this key to retrieve the output information, such as files and logs, in the 'Dashboard' module (**Figure 3c, d, e, f**). All submitted tasks enter the task queue supported by the SQLite database using the R package litseq (https://CRAN.R-project.org/package=liteq). Tasks in the queue are automatically checked by the activated workers (**Figure 3e, f**).

Instant module. The 'Instant' module is used to run the real-time plots and data analysis, and similar to the 'Pipeline' module, the UI and server were automatically generated via plugin configuration files (**Dataset S5**). We used the meta database query of BioInstaller, Datasets2tools (Torre et al. 2018), PubMed, and plots of Maftools (Mayakonda & Koeffler 2016), a cancer somatic mutations visualization tool, as the demo to demonstrate the function. Users can select the input files defined in the plugins configuration file (TOML) or user-uploaded files. The commands are stored in the bottom of the boxes and can be modified by the user. After clicking the 'Run' button, all output box codes, such as output plots and tables, run on the server side and are returned in real time to the Shiny UI (**Figure 4**). We developed several plugins to query and access several meta databases related to bioinformatics, such as the BioInstaller meta databases (**Figure 4a, b**),

Datasets2tools (Torre et al. 2018) (**Figure 4c, d**), and PubMed (**Figure 4e, f**). The powerful visualization functions of R packages are also supported in the 'Instant' module. As shown in **Figure 4g, h,** users can obtain the demo output (PDF and PNG format) of Maftools. After running all box codes, a single box can be separately updated and exported by users.

Installer module. The 'Installer' module is the main Shiny interface of BioInstaller for downloading and installing bioinformatics tools/scripts and databases. We provide the Shiny interfaces of BioInstaller, conda and spack (**Figure S4**). The 'Installer' module is similar to the 'Pipeline' module, which is also needed to submit a task to the queue. The status and log information can be retrieved in the 'Dashboard' module. Three basic use cases of the BioInstaller Shiny application are available: 1) download db_annovar_refgene database (**Figure S4a**); 2) create conda environment (**Figure S4b)**; 3) install 'zlib' using spack **(Figure S4c)**.

Setting module. The setting module is the interface for setting the value of the variable used in the BioInstaller plugins or R files. Both a Shiny UI and a YAML editor are offered for users (**Figure S5a, b**). Any updates of the YAML editor (**Figure S5b**) can change and refresh the Shiny UI options (**Figure S5a**). It is helpful for users to manage various material related to BioInstaller and its plugins.

In most cases, through the one-click interface of BioInstaller, users can easily download and install the desired bioinformatics resources without any command line skills. Functions for automatic compiling from the source file with the dependent software or database are also supported in BioInstaller. However, for complicated software with high system dependence, we recommend using the interfaces of conda (https://conda.io/docs/) and spack (Gamblin et al. 2015).

Portable message queue for background tasks based on SQLite

Tasks with long-time costs are challenging in Shiny, which always blocks the other interactive operations simultaneously when the previous task has not been finished. Here, we utilized the R package litseq (https://CRAN.R-project.org/package=liteq) to submit and manage the background queue tasks. litseq is portable and lightweight. litseq does not require extra software or service from other programming platforms and can work on any clusters server running computing-intensive tasks. The developed queue worker in BioInstaller can be used for all other background tasks submitted by litseq. All litseq-submitted tasks of BioInstaller are assigned a unique identification id. All executed commands, output logs, and others are saved in the permanent files.

Opencpu backend service improves reproducibility

Opencpu (Ooms 2017) is an R package for reproducible research that can expose a web REST API interface with R, Latex and, Pandoc. The R functions of BioInstaller are invoked by the activated Opencpu R process or daemon service. For other programming platform users, this is one possible method for utilizing the R functions of BioInstaller (**Figure 1**). The output of JSON and text formats are returned when using the browser access (**Figure 5a**) or simulated requests. Three of the most basic APIs usages of BioInstaller were used to demonstrate how it works: 1) obtaining all supported tools/scripts and databases; 2) acquiring available versions of the appointed item; 3) installing a tool in a directory (**Figure 5b**). Notably, a random string, such as "x0a469794fa", will be generated as the key of Opencpu to obtain the output of one R session. Both JSON and text format output can be returned by Opencpu backend APIs (**Figure 5c**).

Docker container of BioInstaller

A prebuilt docker image is available on the docker hub (https://hub.docker.com/r/bioinstaller/bioinstaller), and the latest code change of the BioInstaller repository can automatically trigger an update of the docker image. In the docker image, we integrated and configured three types of web services, including Opencpu, Shiny (Chang et al. 2015; Ooms 2017), and the RStudio server (https://www.rstudio.com/products/rstudio-server/). The followed commands can be used to deploy and start the service of BioInstaller service.

```
$ docker pull bioinstaller/bioinstaller
$ docker run -it -p 80:80 -p 8004:8004 bioinstaller/bioinstaller
```

Users can deploy a new instance host of BioInstaller and all other web services in a few minutes, and other tools/scripts and databases are also allowed to be embedded in this docker image using the Dockerfile (https://github.com/JhuangLab/BioInstaller/blob/master/Dockerfile).

Use the GitHub forum to share, rate, and discuss the bioinformatics resources

The full-text search is natively supported by the GitHub website with highlight and age forwarding functions (**Figure S6a, b**). To simplify the submitting of new items to BioInstaller, the GitHub repository issues page (https://github.com/JhuangLab/BioInstaller/issues) is recommended for other users to share, rate, and discuss bioinformatics tools/scripts and databases with a designated

label (**Figure S6c**). The 'watching' function of GitHub can allow users to receive notifications of all conversations on the BioInstaller. Another advantage of establishing a free sharing community based on the GitHub is that all history changes on the code and forum posts can be recorded and retrieved. A rating function for bioinformatics tools/scripts or databases is also feasible by calculating the points corresponding to thumbs up or down.

DISCUSSION

Bioinformatics tools/scripts and databases are widely used in various data analysis projects. The construction of interactive and reproducible biological data analysis applications is critical for most bioinformatics data analyses (Henry et al. 2014; McQuilton et al. 2016; Ohno-Machado et al. 2017). The integrative utilization of these resources is becoming incrreasingly important for improving integrated biosciences data analysis. R language, as the most popular programming language for statistics, biological data analysis, and big data, has provided massive useful R packages for various data analysis efforts, especially the NGS field. However, there has been no comprehensive and free R application that can support file upload and management, perform long-time computation with a tasks submission system, track and record the output of files and log, develop extendible plugins, add or remove functions of the application in real time, and respond for REST APIs. Another common problem for users of R and other programming platforms for biological data analysis is that massive bioinformatics resources are isolated and scattered, which significantly increases the difficulty of deploying, collecting and sharing these resources. Well-known software distribution tools that do not need root privileges, such as conda (https://conda.io/docs) and spack (Gamblin et al. 2015), were designed for comprehensive fields and usually lack support for life science resources. Bioconda is a fine example of the centralized installation of bioinformatics software (approximately 1900 items) that has significantly improved the reproducibility of bioinformatics data analysis (Gruning et al. 2018). However, this is not sufficient compared with the rapid increase in software and databases in the life sciences field.

As described in this study, we present a comprehensive, free and open-source platform, BioInstaller, to construct the interactive and reproducible biological data analysis applications. BioInstaller contains the R functions, the Shiny application, REST APIs and the docker image. This platform and the practices described in this work are sufficient for most R users to conveniently and quickly develop an interactive and reproducible biological data analysis

application with diverse predefined functions (e.g., file management, task submission, plugin management system, logging, etc.), plugins, and files offered by BioInstaller. Moreover, based on the TOML format files, we have also integrated hundreds of bioinformatics resources required for the wide field of bioinformatics, such as sequence alignment, variant calling and annotation, and so on. We hope this newly presented open source platform for R users can reduce the difficulty of constructing the interactive and reproducible biological data analysis applications and further improve the interactivity and reproducibility of bioinformatics data analysis.

CONCLUSION

As described in this work, we established a new platform to construct interactive and reproducible biological data analysis applications based on R language. This platform contains diverse user interfaces, including the R functions and R Shiny applitation, REST APIs, and support for collecting, managing, sharing, and utilizing massive bioinformatics tools/scripts and databases.

ACKNOWLEDGEMENT

We would like to thank the many BioInstaller users who provided feedback and suggestions. We thank the Centre for HPC at Shanghai Jiao Tong University for supporting computing platforms.

REFERENCES

Abecasis GR, Auton A, Brooks LD, DePristo MA, Durbin RM, Handsaker RE, Kang HM, Marth GT, and McVean GA. 2012. An integrated map of genetic variation from 1,092 human genomes. *Nature* 491:56-65. 10.1038/nature11632

Afgan E, Baker D, van den Beek M, Blankenberg D, Bouvier D, Cech M, Chilton J, Clements D, Coraor N, Eberhard C, Gruning B, Guerler A, Hillman-Jackson J, Von Kuster G, Rasche E, Soranzo N, Turaga N, Taylor J, Nekrutenko A, and Goecks J. 2016. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. *Nucleic Acids Res* 44:W3-W10. 10.1093/nar/gkw343

Cancer Genome Atlas Research N, Weinstein JN, Collisson EA, Mills GB, Shaw KR, Ozenberger BA, Ellrott K, Shmulevich I, Sander C, and Stuart JM. 2013. The Cancer Genome Atlas Pan-Cancer analysis project. *Nat Genet* 45:1113-1120. 10.1038/ng.2764

Chang MT, Asthana S, Gao SP, Lee BH, Chapman JS, Kandoth C, Gao J, Socci ND, Solit DB, Olshen AB, Schultz N, and Taylor BS. 2016. Identifying recurrent mutations in cancer reveals widespread lineage diversity and mutational specificity. *Nat Biotechnol* 34:155-163. 10.1038/nbt.3391

Chang W, Cheng J, Allaire J, Xie Y, and McPherson J. 2015. Shiny: web application framework for R. *R package version 011* 1:106.

Consortium GT. 2013. The Genotype-Tissue Expression (GTEx) project. *Nat Genet* 45:580-585. 10.1038/ng.2653

Derrien T, Johnson R, Bussotti G, Tanzer A, Djebali S, Tilgner H, Guernec G, Martin D, Merkel A, Knowles DG, Lagarde J, Veeravalli L, Ruan X, Ruan Y, Lassmann T, Carninci P, Brown JB, Lipovich L, Gonzalez JM, Thomas M, Davis CA, Shiekhattar R, Gingeras TR, Hubbard TJ, Notredame C, Harrow J, and Guigo R. 2012. The GENCODE v7 catalog of human long noncoding RNAs: analysis of their gene structure, evolution, and expression. *Genome Res* 22:1775-1789. 10.1101/gr.132159.111

Gamblin T, LeGendre M, Collette MR, Lee GL, Moody A, de Supinski BR, and Futral S. 2015. The Spack package manager: bringing order to HPC software chaos. Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis: ACM. p 40.

Gentleman RC, Carey VJ, Bates DM, Bolstad B, Dettling M, Dudoit S, Ellis B, Gautier L, Ge Y, Gentry J, Hornik K, Hothorn T, Huber W, Iacus S, Irizarry R, Leisch F, Li C, Maechler M, Rossini AJ, Sawitzki G, Smith C, Smyth G, Tierney L, Yang JY, and Zhang J. 2004. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol* 5:R80. 10.1186/gb-2004-5-10-r80

Griffith M, Spies NC, Krysiak K, McMichael JF, Coffman AC, Danos AM, Ainscough BJ, Ramirez CA, Rieke DT, Kujan L, Barnell EK, Wagner AH, Skidmore ZL, Wollam A, Liu CJ, Jones MR, Bilski RL, Lesurf R, Feng YY, Shah NM, Bonakdar M, Trani L, Matlock M, Ramu A, Campbell KM, Spies GC, Graubert AP, Gangavarapu K, Eldred JM, Larson DE, Walker JR, Good BM, Wu C, Su AI, Dienstmann R, Margolin AA, Tamborero D, Lopez-Bigas N, Jones SJ, Bose R, Spencer DH, Wartman LD, Wilson RK, Mardis ER, and Griffith OL. 2017. CIViC is a community knowledgebase for expert crowdsourcing the clinical interpretation of variants in cancer. *Nat Genet* 49:170-174. 10.1038/ng.3774

Gruning B, Dale R, Sjodin A, Chapman BA, Rowe J, Tomkins-Tinch CH, Valieris R, Koster J, and Bioconda T. 2018. Bioconda: sustainable and comprehensive software distribution for the life sciences. *Nat Methods* 15:475-476. 10.1038/s41592-018-0046-7

Henry VJ, Bandrowski AE, Pepin AS, Gonzalez BJ, and Desfeux A. 2014. OMICtools: an informative directory for multi-omic data analysis. *Database,2014,(2014-01-01)* 2014:2091-2092.

Huether R, Dong L, Chen X, Wu G, Parker M, Wei L, Ma J, Edmonson MN, Hedlund EK, Rusch MC, Shurtleff SA, Mulder HL, Boggs K, Vadordaria B, Cheng J, Yergeau D, Song G, Becksfort J, Lemmon G, Weber C, Cai Z, Dang J, Walsh M, Gedman AL, Faber Z, Easton J, Gruber T, Kriwacki RW, Partridge JF, Ding L, Wilson RK, Mardis ER, Mullighan CG, Gilbertson RJ, Baker SJ, Zambetti G, Ellison DW, Zhang J, and Downing JR. 2014. The landscape of somatic mutations in epigenetic regulators across 1,000 paediatric cancer genomes. *Nat Commun* 5:3630. 10.1038/ncomms4630

Mayakonda A, and Koeffler HP. 2016. Maftools: Efficient analysis, visualization and summarization of MAF files from large-scale cohort based cancer studies. *Available at* https://www.biorxiv.org/content/biorxiv/early/2016/05/11/052662.full.pdf (accessed Sep.3 2018).

McLaren W, Gil L, Hunt SE, Riat HS, Ritchie GR, Thormann A, Flicek P, and Cunningham F. 2016. The Ensembl Variant Effect Predictor. *Genome Biol* 17:122. 10.1186/s13059-016-0974-4

McQuilton P, Gonzalez-Beltran A, Rocca-Serra P, Thurston M, Lister A, Maguire E, and Sansone SA. 2016. BioSharing: curated and crowd-sourced metadata standards, databases and data policies in the life sciences. *Database (Oxford)* 2016. 10.1093/database/baw075

O'Leary NA, Wright MW, Brister JR, Ciufo S, Haddad D, McVeigh R, Rajput B, Robbertse B, Smith-White B, Ako-Adjei D, Astashyn A, Badretdin A, Bao Y, Blinkova O, Brover V, Chetvernin V, Choi J, Cox E, Ermolaeva O, Farrell CM, Goldfarb T, Gupta T, Haft D, Hatcher E, Hlavina W, Joardar VS, Kodali VK, Li W, Maglott D, Masterson P, McGarvey KM, Murphy MR, O'Neill K, Pujar S, Rangwala SH, Rausch D, Riddick LD, Schoch C, Shkeda A, Storz SS, Sun H, Thibaud-Nissen F, Tolstoy I, Tully RE, Vatsan AR, Wallin C, Webb D, Wu W, Landrum MJ, Kimchi A, Tatusova T, DiCuccio M, Kitts P,

Murphy TD, and Pruitt KD. 2016. Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Res* 44:D733-745. 10.1093/nar/gkv1189

Ohno-Machado L, Sansone SA, Alter G, Fore I, Grethe J, Xu H, Gonzalez-Beltran A, Rocca-Serra P, Gururaj AE, Bell E, Soysal E, Zong N, and Kim HE. 2017. Finding useful data across multiple biomedical data repositories using DataMed. *Nat Genet* 49:816-819. 10.1038/ng.3864

Ooms J. 2017. opencpu: Producing and Reproducing Results. *Available at https://CRAN.R-project.org/package=opencpu* (accessed Sep. 3 2018).

Pedersen BS, Layer RM, and Quinlan AR. 2016. Vcfanno: fast, flexible annotation of genetic variants. *Genome Biol* 17:118. 10.1186/s13059-016-0973-5

Piñero J, Bravo À, Queralt-Rosinach N, Gutiérrez-Sacristán A, Deu-Pons J, Centeno E, García-García J, Sanz F, and Furlong LI. 2017. DisGeNET: a comprehensive platform integrating information on human disease-associated genes and variants. *Nucleic Acids Research* 45:D833-D839. 10.1093/nar/gkw943

Ramos AH, Lichtenstein L, Gupta M, Lawrence MS, Pugh TJ, Saksena G, Meyerson M, and Getz G. 2015. Oncotator: Cancer Variant Annotation Tool. *Human Mutation* 36:2423-2429.

Sanchez-Vega F, Mina M, Armenia J, Chatila WK, Luna A, La KC, Dimitriadoy S, Liu DL, Kantheti HS, Saghafinia S, Chakravarty D, Daian F, Gao Q, Bailey MH, Liang WW, Foltz SM, Shmulevich I, Ding L, Heins Z, Ochoa A, Gross B, Gao J, Zhang H, Kundra R, Kandoth C, Bahceci I, Dervishi L, Dogrusoz U, Zhou W, Shen H, Laird PW, Way GP, Greene CS, Liang H, Xiao Y, Wang C, Iavarone A, Berger AH, Bivona TG, Lazar AJ, Hammer GD, Giordano T, Kwong LN, McArthur G, Huang C, Tward AD, Frederick MJ, McCormick F, Meyerson M, Cancer Genome Atlas Research N, Van Allen EM, Cherniack AD, Ciriello G, Sander C, and Schultz N. 2018. Oncogenic Signaling Pathways in The Cancer Genome Atlas. *Cell* 173:321-337 e310. 10.1016/j.cell.2018.03.035

Torre D, Krawczuk P, Jagodnik KM, Lachmann A, Wang Z, Wang L, Kuleshov MV, and Ma'ayan A. 2018. Datasets2Tools, repository and search engine for bioinformatics datasets, tools and canned analyses. *Sci Data* 5:180023. 10.1038/sdata.2018.23

Tyner C, Barber GP, Casper J, Clawson H, Diekhans M, Eisenhart C, Fischer CM, Gibson D, Gonzalez JN, Guruvadoo L, Haeussler M, Heitner S, Hinrichs AS, Karolchik D, Lee BT, Lee CM, Nejad P, Raney BJ, Rosenbloom KR, Speir ML, Villarreal C, Vivian J, Zweig AS, Haussler D, Kuhn RM, and Kent WJ. 2017. The UCSC Genome Browser database: 2017 update. *Nucleic Acids Res* 45:D626-D634. 10.1093/nar/gkw1134

Wala JA, Bandopadhayay P, Greenwald N, O'Rourke R, Sharpe T, Stewart C, Schumacher S, Li Y, Weischenfeldt J, Yao X, Nusbaum C, Campbell P, Getz G, Meyerson M, Zhang CZ, Imielinski M, and Beroukhim R. 2018. SvABA: genome-wide detection of structural variants and indels by local assembly. *Genome Res*. 10.1101/gr.221028.117

Wang K, Li M, and Hakonarson H. 2010. ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. *Nucleic Acids Res* 38:e164. 10.1093/nar/gkq603

Zook JM, Chapman B, Wang J, Mittelman D, Hofmann O, Hide W, and Salit M. 2014. Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat Biotechnol* 32:246-251. 10.1038/nbt.2835

Figure 1(on next page)

Overview of structure and functions of BioInstaller.

Bioinformatics tools, scripts and databases are supported by BioInstaller. Bootstrap and Shinydashbord are used to construct the front-end interface. The R functions, Shiny and Opencpu services and the SQLite and TOML databases were applied in the back-end.

Client

Browser

BioInstaller

Cloud computing toolkit

- ✓Files upload and management
- ✓Export plots and tables
- ✓Instant rendering
- ✓Task submission
- ✓Task queue and worker
- ✓Monitor of device and tasks
- ✓Logging
- ✓Extendible plugins

.....

Front-end interface

Bootstrap + Shinydashbord

Server-side

File system

SQLite / TOML

R functions

install.bioinfo
install.github
Install.nongithub
crawl.all.versions
get.meta

show.installed
new.bioinfo
web
conda
spack

.....

R users

Other programming platform

(TEXT + JSON)

Tool | Script | Database

Configuration file pool (TOML files)

Interfaces of BioInstaller

R | Shiny | APIs

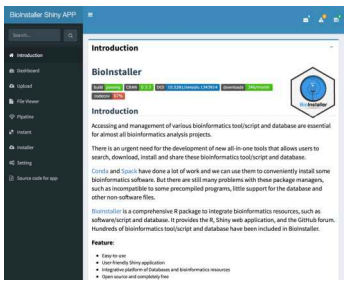
Code library and plugins

Collect | Manage | Share | Utilze

Public and internal usage

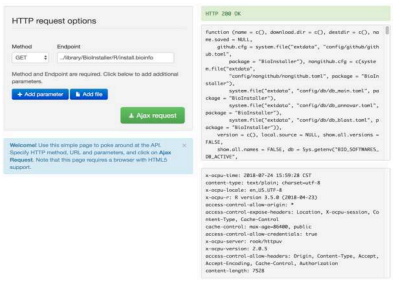
R and Shiny

RStudio Shiny



Opencpu

Opencpu (GET/POST)



Interactive and reproducible biological data analysis

Figure 2(on next page)

The relevance, applicability and a real example of BioInstaller

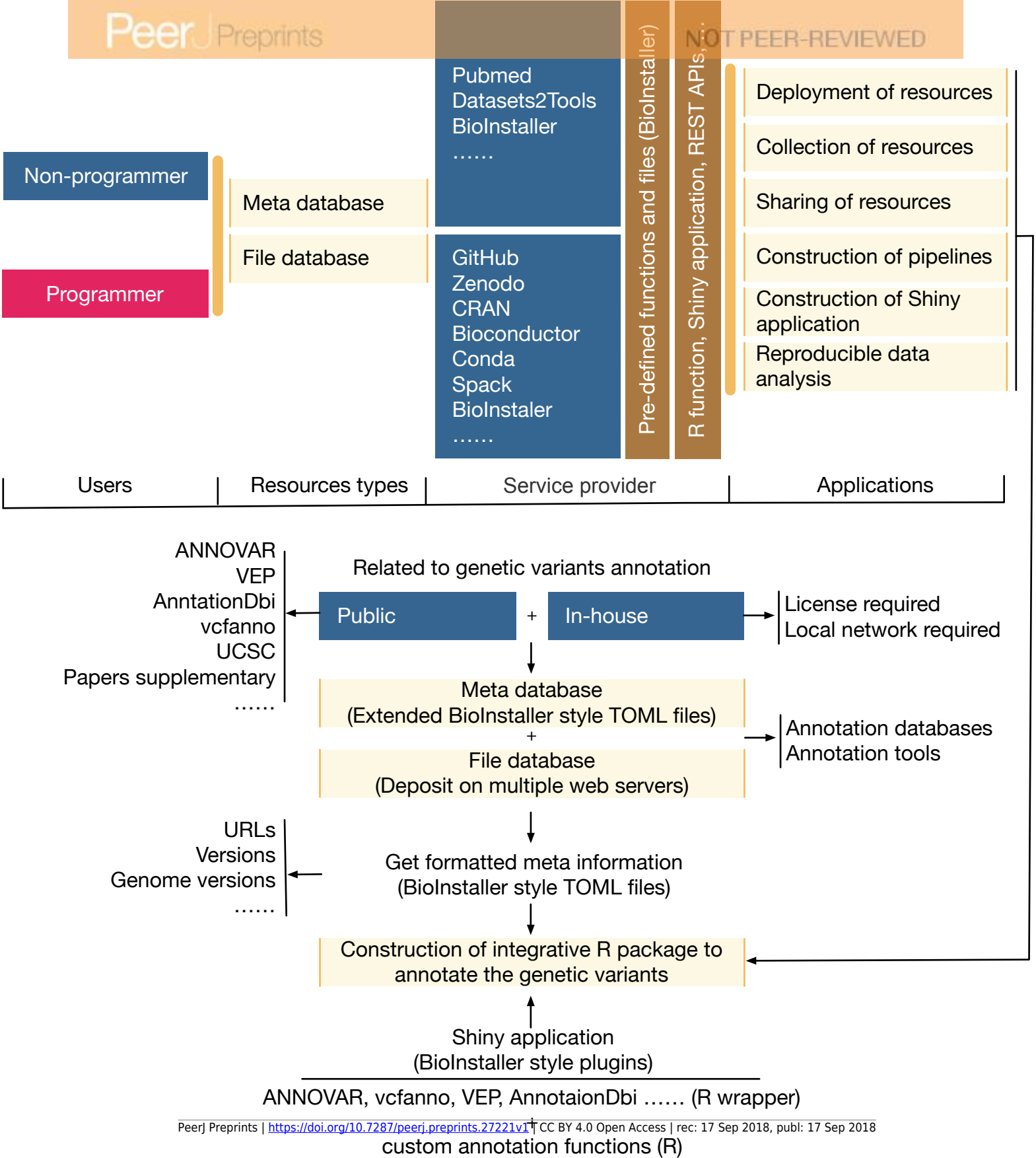


Figure 3(on next page)

Shiny application modules of file viewer and pipeline.

(A) Uploaded files are showed on the table where view, deletion, and download function are provided. (B) The interface of the preview result. (C) Easy project was used as the demo in pipeline module, which could be used to create a series of directories via submitting a queue task with two parameters: project name and parent directory. (D) The dialog box displays a prompt message with a queue character key. (E) Task queue and queue information can be requested by the character key in the dashboard module. (F) Function to get the output log of the submitted task.

A

File List

Copy CSV Excel PDF Print Search:

All files stored in shinyapp Web service

ID	Action	file_name	file_path	file_size	file
11		rap-2pg-cv.pdf	/Users/ljf/.annovarR/upload/11	91516	pdf
12		ir-15-546.pdf	/Users/ljf/.annovarR/upload/12	1666944	pdf
13		rap-2pg-cv.pdf	/Users/ljf/.annovarR/upload/13	91516	pdf
14		dat.txt	/Users/ljf/.BioInstaller/upload/14	66500	txt

Show 5 entries

Showing 11 to 14 of 14 entries Previous 1 2 3 Next

B

File Preview

Copy CSV Excel PDF Print Search:

All files stored in shinyapp Web service

date	EMS	neonatal	infant	kid	child	adolescent	youth	middle age
20160101	426	0	0	7	0	6	61	4.
20160102	424	0	1	4	1	4	55	3.
20160103	393	0	1	0	1	0	47	4.
20160104	420	0	0	1	0	3	52	4.
20160105	397	0	2	3	0	1	46	4.

Show 5 entries

Showing 1 to 5 of 366 entries Previous 1 2 3 4 5 … 74 Next

C

easy_project

New bioinformatics analysis project

Parameters:

Project Name

Project parent dir

Submit

R commands for task

```
parent_dir <- normalizePath(parent_dir, mustWork = FALSE)
project_dir <- normalizePath(file.path(parent_dir, project_name), mustWork
analysis_dirs <- c(paste0("rnaseq/", c("exp", "fusion", "splicing", "mutat
                  paste0("dnaseq/", c("wes/mutation", "wgs/mutation", "ch
data_dirs <- c("fastq", "fasta", "bam", "vcf", "meta")
meta_json <- list(project_dir = project_dir, project_name = project_name,
  create_time = format(Sys.time(), "%Y-%m-%d %H:%M:%S"),
creator = Sys.getenv("USER"))
meta_json$project_id <- git2r::hash(sprintf("%s,%s",
                                meta_json$project_name, meta_json$crea
sapply(sprintf("%s/%s", project_dir, c(file.path("analysis", analysis_dirs
                                  file.path("data", data_dirs))),
                                  function(x) {dir.create(x, recursiv
configr::write.config(meta_json, file.path(project_dir, "project.json"), w
```

D

Submit success!

Please copy and backup the followed messages. It is required to access the output or run in your command line client.

You can access the result via input key in the dashbord page:

```
SOuG5jOkN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I
```

Commands and paramters:


```
{"input2var":{"project_name":["input_project_name"],"parent_dir":
["input_parent_dir"]},"input":{"input_project_name":["B-ALL"],"in
put_parent_dir":["/tmp"]},"req_pkgs":[""],"qqcommand":[""],"qqkey
":["SQuG5jOkN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I"],"qqcomma
nd_type":["R"],"boxes":["new_proj"],"last_cmd":["parent_dir <- no
rmalizePath(parent_dir, mustWork = FALSE)\nproject_dir <- normali
zePath(file.path(parent_dir, project_name), mustWork = FALSE)\nan
alysis_dirs <- c(paste0(\"rnaseq/\", c(\"exp\", \"fusion\", \"spl
icing\", \"mutation\")),\n                  paste0(\"dnaseq/\",
c(\"wes/mutation\", \"wgs/mutation\", \"chip/peak\")))\ndata_dirs
 <- c(\"fastq\", \"fasta\", \"bam\", \"vcf\", \"meta\")\nmeta_jso
n <- list(project_dir = project_dir, project_name = project_name,
\n  create_time = format(Sys.time(), \"%Y-%m-%d %H:%M:%S\"),\ncre
ator = Sys.getenv(\"USER\"))\nmeta_json$project_id <- git2r::hash
(sprintf(\"%s,%s\",\n                                    meta_jso
n$project_name, meta_json$create_time))\nsapply(sprintf(\"%s/%s\"
, project_dir, c(file.path(\"analysis\", analysis_dirs),\n
                                file.path(\"data\", data_dirs))),
\n                                    function(x) {dir.create(
x, recursive = TRUE)})\nconfigr::write.config(meta_json, file.pat
h(project_dir, \"project.json\"), write.type = \"json\")\n"]}
```


E

Task table query

Key

SQuG5jOkN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I

Query

Task information

Copy CSV Excel PDF Print Search:

Task information

id	msgid	key	status	log
98	106	SQuG5jOkN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I	FINISHED	~/.BioInst

Show 5 entries

Showing 1 to 1 of 1 entries Previous 1 Next

F

Output of log

```
---------------------- ~/.BioInstaller/log/SQuG5jOkN355S1J0FL6ZIOHds9l0viefBa43otseYa3unhRS9I
.log ----------------------
project_name =>
[1] "B-ALL"
parent_dir =>
[1] "/tmp"
parent_dir <- normalizePath(parent_dir, mustWork = FALSE)
project_dir <- normalizePath(file.path(parent_dir, project_name), mustWork = FALSE)
analysis_dirs <- c(paste0("rnaseq/", c("exp", "fusion", "splicing", "mutation")),
                   paste0("dnaseq/", c("wes/mutation", "wgs/mutation", "chip/peak")))
data_dirs <- c("fastq", "fasta", "bam", "vcf", "meta")
meta_json <- list(project_dir = project_dir, project_name = project_name,
  create_time = format(Sys.time(), "%Y-%m-%d %H:%M:%S"),
creator = Sys.getenv("USER"))
meta_json$project_id <- git2r::hash(sprintf("%s,%s",
                                    meta_json$project_name, meta_json$create_time))
sapply(sprintf("%s/%s", project_dir, c(file.path("analysis", analysis_dirs),
                                       file.path("data", data_dirs))),
                                       function(x) {dir.create(x, recursive = TRUE)})
configr::write.config(meta_json, file.path(project_dir, "project.json"), write.type = "json")
```

Figure 4(on next page)

Shiny application 'Instant' module

(A)(B) A demonstration show how Shiny 'Installer' module works by downloading 'db_ucsc_refgene'. Dynamic and interactive manipulations are supported. Log information of submitted download/install job can be recalled using given random characters. (C)(D) The input box and output log when a new softwares environment is created by the conda plugin of 'Installer' module. (E)(F) The input box and output log when 'zlib' is installed by the spack plugin of 'Installer' module.

A

bioinstaller | spack | conda | datasets2tools | pubmed | geoquery | maftools

BioInstaller meta database query

Query type

github

Run

C

Dataset2tools meta database query

Query type

tool

Key Words

Dataset Accession (e.g. GSE775)

Tool name

ANNOVAR

Disease name

Canned nalysis Cccession

Geneset

Counts

100

Run

E

Pubmed query

Term (e.g. 30105797,30105798,B-ALL)

vcfanno

Counts

100

Pubmed Account Key

193124979d2e7f360c150dadc5b1e3bfec09

Run

G

Input files

Single sample anlayais:

MAF file (laml_maf)

/Library/Frameworks/R.framework/Versions/3.5/Resources/library/maftools/extdata/tcga_laml.maf.gz

Clincal annotation file (laml_clin)

/Library/Frameworks/R.framework/Versions/3.5/Resources/library/maftools/extdata/tcga_laml_annot.tsv

MutSig reusults (laml_mutsig)

/Library/Frameworks/R.framework/Versions/3.5/Resources/library/maftools/extdata/LAML_sig_genes.txt.gz

Other paramters

Reading file: Maf file is TCGA format? (is_tcga is TRUE)

Run

R commands for task

```
laml = read.maf(maf = laml_maf, clinicalData = laml_clin)
laml.plus.gistic = read.maf(maf = laml_maf, gisticAllLesionsFile = all_lesions, gisticAmpGenesFile = amp_genes,
  gisticDelGenesFile = del_genes, gisticScoresFile = scores_gis, isTCGA = is_tcga)
laml.gistic = readGistic(gisticAllLesionsFile = all_lesions, gisticAmpGenesFile = amp_genes, gisticDelGenesFile = del_genes, gisticScoresFile
= scores_gis, isTCGA = is_tcga)
primary_maf = read.maf(maf = primary_maf)
relapse_maf = read.maf(maf = relapse_maf)
```

B

Output of bioinstaller (Table)

Copy | CSV | Excel | PDF | Print

Search: vcfanno

Output table

Name	Description	Publication	resources_db
vcfanno	vcfanno allows you to quickly annotate your VCF with any number of INFO fields from any number of VCFs or BED files. It uses a simple conf file to allow the user to specify the source annotation files and fields and how they will be added to the info of the query VCF.	Pedersen B S, Layer R M, Quinlan A R. Vcfanno: fast, flexible annotation of genetic variants[J]. Genome Biology, 2016, 17(1):1-9.	nongithub

Show 5 entries

Showing 1 to 1 of 1 entries (filtered from 115 total entries)

Previous 1 Next

D

Output of Datasets2tools (Table)

Copy | CSV | Excel | PDF | Print

Search: ANNOVAR

Output table

	analyses	tool_description	articles	tool_name	tool_accession
0		Functional annotation of genetic variants from high-throughput sequencing data	https://doi.org/10.1093/nar/gkq603	ANNOVAR	DCT00003217

Show 5 entries

Showing 1 to 1 of 1 entries

Previous 1 Next

F

Output of pubmed (Table)

Copy | CSV | Excel | PDF | Print

Search: vcfanno

Output table

PMID	DOI	Title	Abstract	PublishDate	JournalName
27250555	10.1186/s13059-016-0973-5	Vcfanno: fast, flexible annotation of genetic variants.	The integration of genome annotations is critical to the identification of genetic variants that are relevant to studies of disease or other traits. However, comprehensive variant annotation with diverse file formats is difficult with existing methods. Here we describe vcfanno, which flexibly extracts and summarizes attributes from multiple annotation files and integrates the annotations within the INFO column of the original VCF file. By leveraging a parallel "chromosome sweeping" algorithm, we demonstrate substantial performance gains by annotating ~85,000 variants per second with 50 attributes from 17 commonly used genome annotation resources. Vcfanno is available at https://github.com/brentp/vcfanno under the MIT license.	20160601	Genome biology

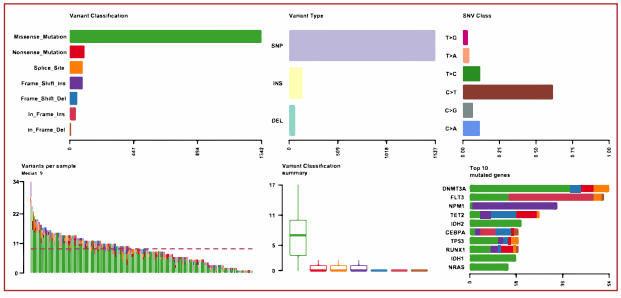
Show 5 entries

Showing 1 to 1 of 1 entries

Previous 1 Next

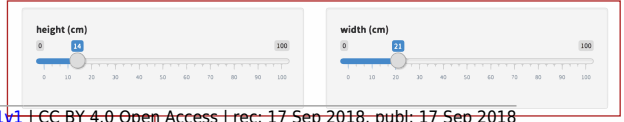
H

Output of maftools MAF summary



R commands for task

```
plotmafSummary(maf = laml, rmOutlier = TRUE, addStat = 'median', dashboard = TRUE, titvRaw = FALSE)
```



Export | Update plot

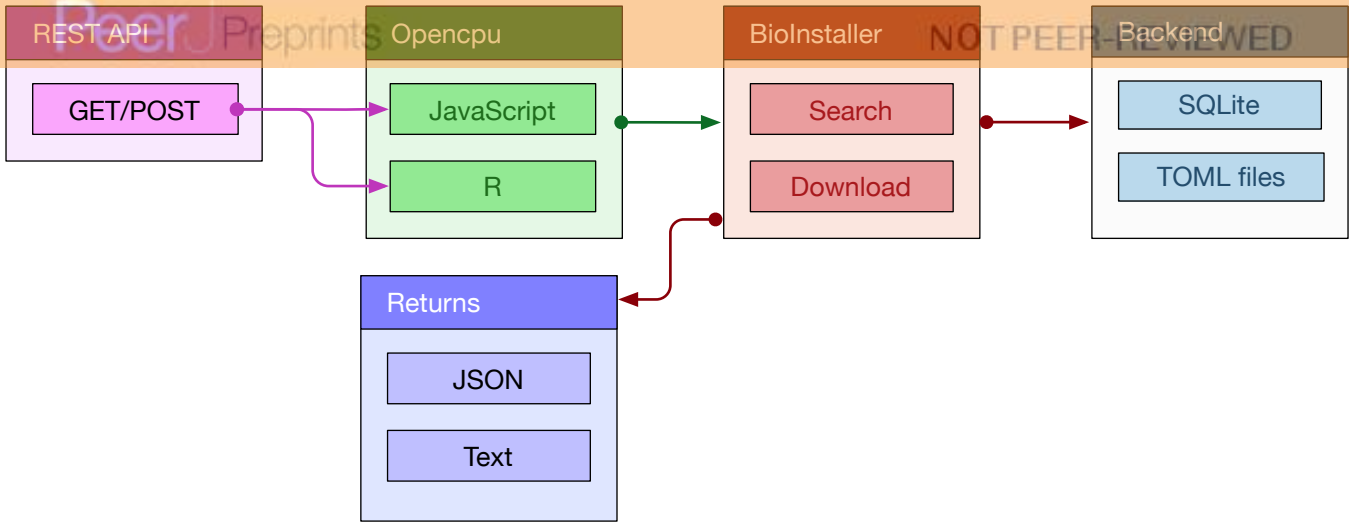
PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.27221v1 | CC BY 4.0 Open Access | rec: 17 Sep 2018, publ: 17 Sep 2018

Figure 5(on next page)

REST APIs of BioInstaller.

(A) Workflow of REST APIs of BioInstaller that JSON and TEXT returns through the GET/POST query. (B) Using curl to invoke background R functions of BioInstaller. (C) The key character with GET method is provided to get the background R session output.

A



B

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo -d \ "show.all.names=TRUE" X POST

Show all items supported by BioInstaller

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo \ -d "name='bwa', show.all.versions=TRUE" -X POST

Show all versions of Bwa

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo \ -d "name='bwa', destdir='/opt/aliner/bwa'" -X POST

Download and install latest Bwa

curl http://localhost:5656/ocpu/library/BioInstaller/R/install.bioinfo \ -d "name='db_annovar_1000g', destdir='/opt/annovardb', \ extra.list=list(buildver='hg19')" -X POST

Download 1000 Genome Project annotation database

C

curl http://localhost:5656/ocpu/tmp/{key}/R/.val/json

Get the JSON format value of returned output

curl http://localhost:5656/ocpu/tmp/{key}/R/.val/text

Get the text format value of returned output

curl http://localhost:5656/ocpu/tmp/{key}/R/.val/print

Get the function 'print' output of returned value

Table 1(on next page)

List of the relevance and applicability of BioInstaller

	With BioInstaller	Without BioInstaller
Deployment of resources		
User-interfaces	R functions, Shiny UI, REST APIs (Conda, Spack, and other tools/scripts)	Command-line tools (Conda, Spack, and custom tools)
Retrieve installed packages	Integrated Shiny dashboard page including R packages, conda and Python packages, Spack packages, and BioInstaller resource	Multiple command line operations
Collection of resources		
Local development	Yes	No
Need to register an account	Not need	Need
Type of backend databases	Default use TOML and SQLite (potable purpose) Plugins for other types	MySQL
Resources hosts	No limitation	Centralized
File sizes	No limitation	Limited
PubMed query	Integrated R codes with secret key (no limited access) Shiny UI with formatted table	Isolated R codes without secret key (limited access, n<=20) Online version without formatted table
Sharing of resources		
Medium	Simplified TOML format files	Form or configuration file required more skills
Download service	Local Shiny application	Centralized web service or command line tools
Construction of pipelines		
Store of meta information (e.g. URL and version)	Pre-defined TOML file	De novo source code (e.g. ANNOVAR and fusioncatcher)
Construction of Shiny application		

Pre-defined pages	Pre-defined Shiny UI and server (Dashboard, file management, task submission, logging, export and update of plots exception handling, setting)	Isolated examples UI and server codes
Difficulty	Easy to construct the Shiny application （Plugins + optional R codes）	Relatively complicated (Require R codes for UI and server)
Reproducible data analysis		
Logging	Support	Manual
Docker image	Pre-defined docker image with Shiny, Rstudio, and Opencpu services	Most not

1

Table 2(on next page)

Overview comparison of BioInstaller and existing tools on the collection and share of bioinformatics resources

	BioInstaller	Omictools	Datasets2Tools
Infrastructure and Utilities			
Programing language	R, JavaScript	HTML/CSS/JavaScript	HTML/CSS/JavaScript
Chrome extension	No	No	Yes
Web service	R Shiny	Web	Web
R functions	Yes	No	No
REST APIs	Yes	No	Yes
Backend database	TOML and SQLite	Not available	MySQL
Docker image	Yes	No	No
Functionality			
Access and collect meta database	Yes	Yes	Yes
Access and collect file database	Yes	No	No
Integration of external resources	Yes	No	No
PubMed query	Yes	No	No
Dataset query	Yes	No	Yes
Number of supported resources	Integrated	High	Medium
Version query	Yes	No	No
Download service	Yes	No	No
Local branch and development	Yes	No	No
Input and output			
Input	R functions, Web text, APIs	Web text only	Web text + APIs
Output	Text, table, plots, and Web page (PNG, SVG and PDF)	Web page	Text and Web page

1

Table 3(on next page)

Summary of BioInstaller included tools/scripts and databases

Category	Number
1, Tools/scripts	
Alignment and assembly	27
Quality control	17
HTS manipulation	17
Association analysis	6
Genetic variants annotation	12
Detection of SNVs, INDELs and SVs	32
Immunity-associated	2
Isoform analysis	3
Gene expression analysis	9
Network analysis	3
Visualization libraries	11
System dependence	18
2, Databases	
Variant-level	
Allele frequency	17
Variants Effect prediction	29
Disease-related	6
Gene-level	
Basic information	8
Gene function	3
Disease-related	7
Drug related	4
Noncoding RNA related	15
Reference genome	9
Protein related	4
Others	8

1