epitweetr: user documentation

European Centre for Disease Prevention and Control (ECDC)

Description

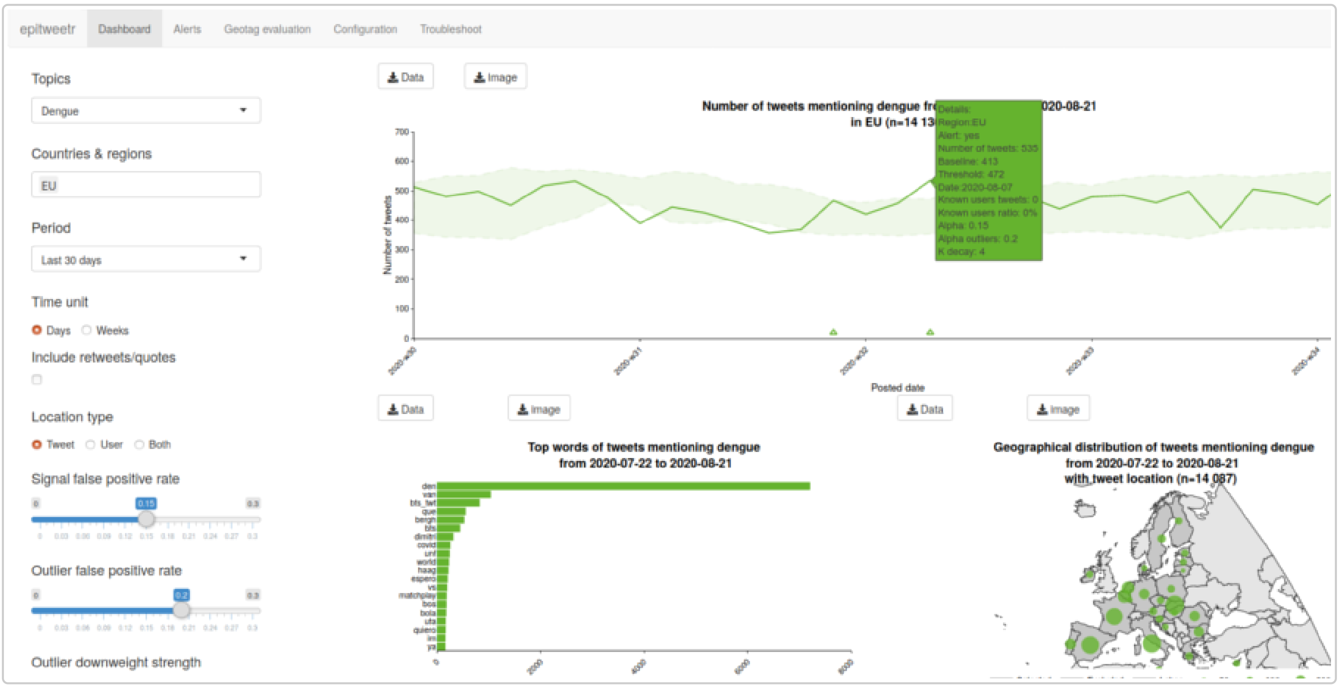
The `epitweetr` package allows you to automatically monitor trends of tweets by time, place and topic. This automated monitoring aims at early detecting public health threats through the detection of signals (e.g. an unusual increase in the number of tweets for a specific time, place and topic). The `epitweetr` package was designed to focus on infectious diseases, and it can be extended to all hazards or other fields of study by modifying the topics and keywords.

The general principle behind `epitweetr` is that it collects tweets and related metadata from the Twitter Standard Search API version 1.1 (https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/overview/standard) according to specified topics and stores these tweets in a compressed form on your computer. `epitweetr` geolocalises the tweets and collects information on key words within a tweet. Tweets are aggregated according to topic and geographical location. Next, a signal detection algorithm identifies the number of tweets (by topic and geographical location) that exceeds what is expected for a given day. Then, `epitweetr` sends out email alerts to notify those who need to further investigate these signals following the epidemic intelligence processes (filtering, validation, analysis and preliminary assessment).

The package includes an interactive web application (Shiny app) with five pages: the dashboard, where a user can visualise and explore tweets (Fig 1), the alerts page, where you can view the current alerts and associated information (Fig 2), the geotag evaluation page, where you can evaluate the geolocation algorithm in different tweet fields to manually choose the geolocation threshold (Fig 3), the configuration page, where you can change settings and check the status of the underlying processes (Fig 4), and the troubleshoot page, with automatic checks and hints for using `epitweetr` with all its functionalities (Fig 5). On the dashboard, users can view the aggregated number of tweets over time, the location of these tweets on a map and the words most frequently found in these tweets. These visualisations can be filtered by the topic, location and time period you are interested in. Other filters are available and include the possibility to adjust the time unit of the timeline, whether retweets/quotes should be included, what kind of geolocation types you are interested in, the sensitivity of the prediction interval for the signal detection, and the number of days used to calculate the threshold for signals. This information is also downloadable directly from this interface in the form of data, pictures, or reports.

Shiny app dashboard:

Fig 1: Shiny app dashboard figure


epitweetr
Dashboard
Alerts
Geotag evaluation
Configuration
Troubleshoot
Data
Image
Number of tweets mentioning dengue fr… 020-08-21 in EU (n=14 13…
Number of tweets
Posted date
Top words of tweets mentioning dengue from 2020-07-22 to 2020-08-21
Geographical distribution of tweets mentioning dengue from 2020-07-22 to 2020-08-21 with tweet location (n=14 087)

Topics: Dengue

Countries & regions: EU

Period: Last 30 days

Time unit: [✓ Days] Weeks

Include retweets/quotes

Location type: [✓ Tweet] User Both

Signal false positive rate: 0.15

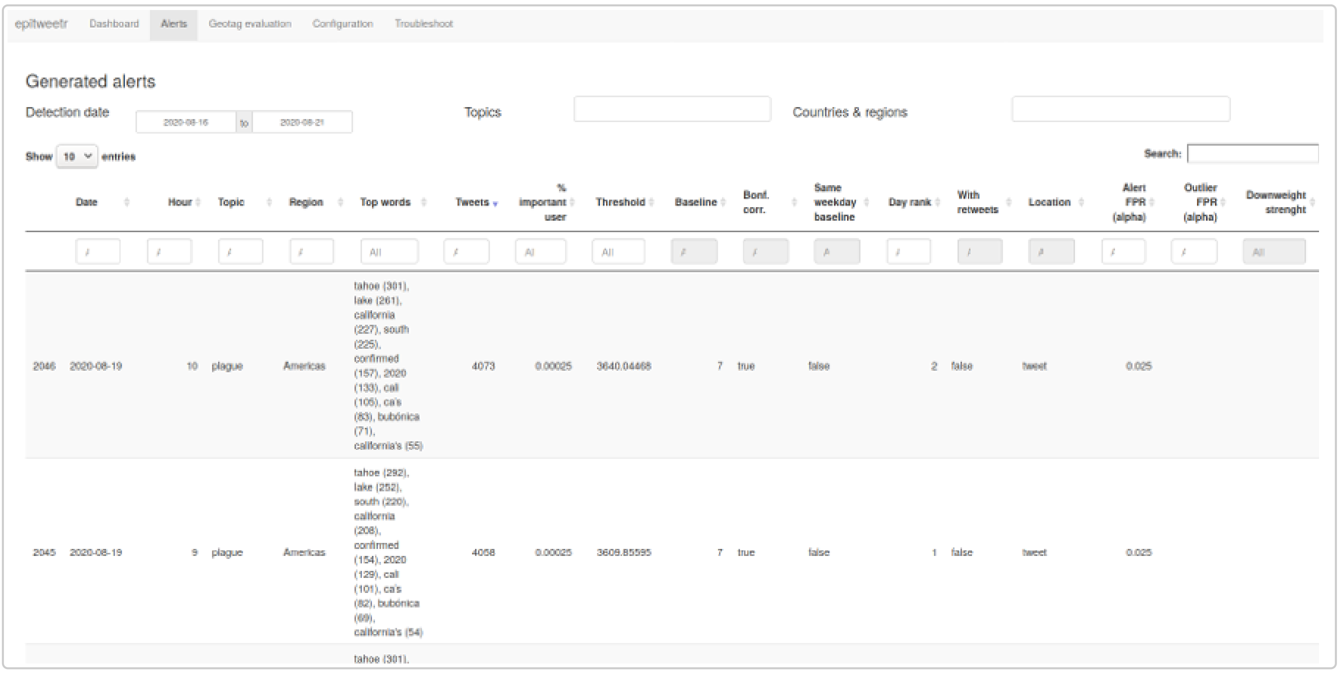
Outlier false positive rate: 0.2

Outlier downweight strength

Details:
Region:EU
Alert: yes
Number of tweets: 535
Baseline: 413
Threshold: 472
Date 2020-08-07
Known users tweets: 0
Known users ratio: 0%
Alpha: 0.15
Alpha outliers: 0.2
K decay: 4

Shiny app alerts page:

Fig 2: Shiny app alerts page



Generated alerts

Detection date: 2020-08-16 to 2020-08-21

Topics

Countries & regions

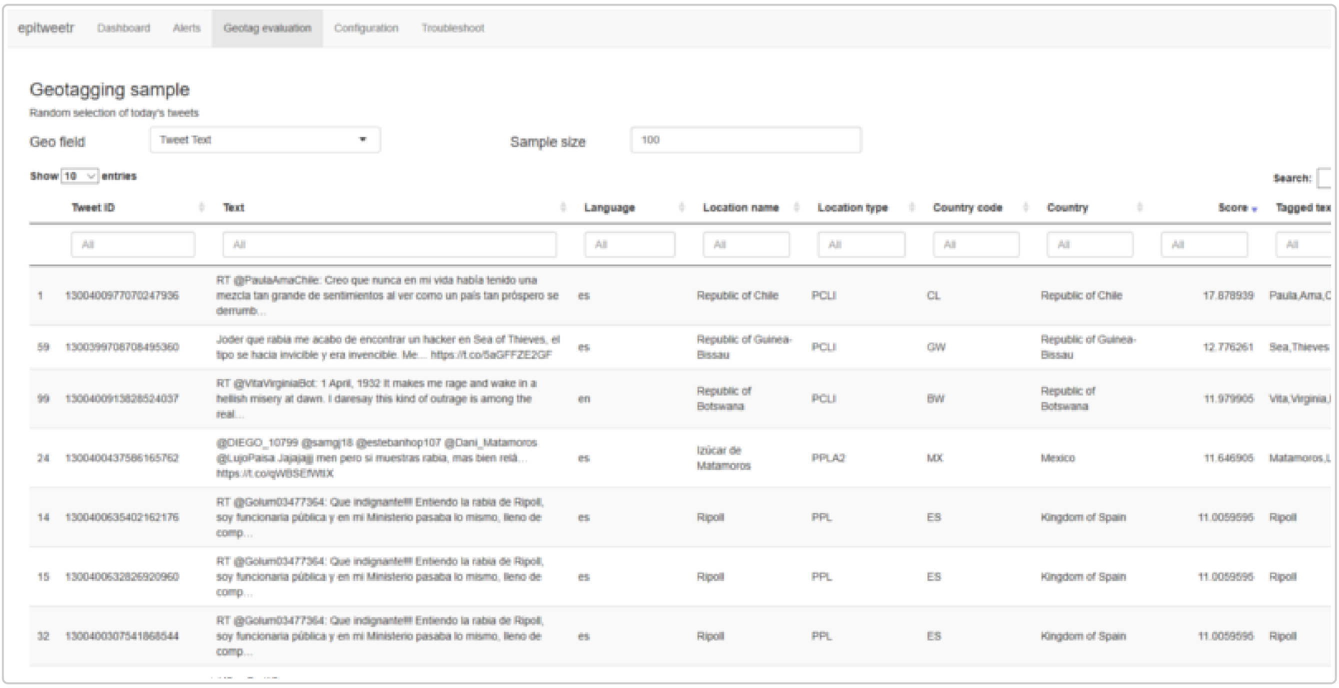
Show 10 entries

Search:

	Date	Hour	Topic	Region	Top words	Tweets	% important user	Threshold	Baseline	Bonf. corr.	Same weekday baseline	Day rank	With retweets	Location	Alert FPR (alpha)	Outlier FPR (alpha)	Downweight strenght
2046	2020-08-19	10	plague	Americas	tahoe (301), lake (261), california (227), south (225), confirmed (157), 2020 (133), cali (105), ca's (83), bubónica (71), california's (55)	4073	0.00025	3640.04468	7	true	false	2	false	tweet	0.025		
2045	2020-08-19	9	plague	Americas	tahoe (292), lake (252), south (220), california (208), confirmed (154), 2020 (129), cali (101), ca's (82), bubónica (69), california's (54)	4058	0.00025	3609.85595	7	true	false	1	false	tweet	0.025		
					tahoe (301),												

Shiny app geotag evaluation page:

Fig 3: Shiny app geotag evaluation page



Geotagging sample

Random selection of today's tweets

Geo field: Tweet Text

Sample size: 100

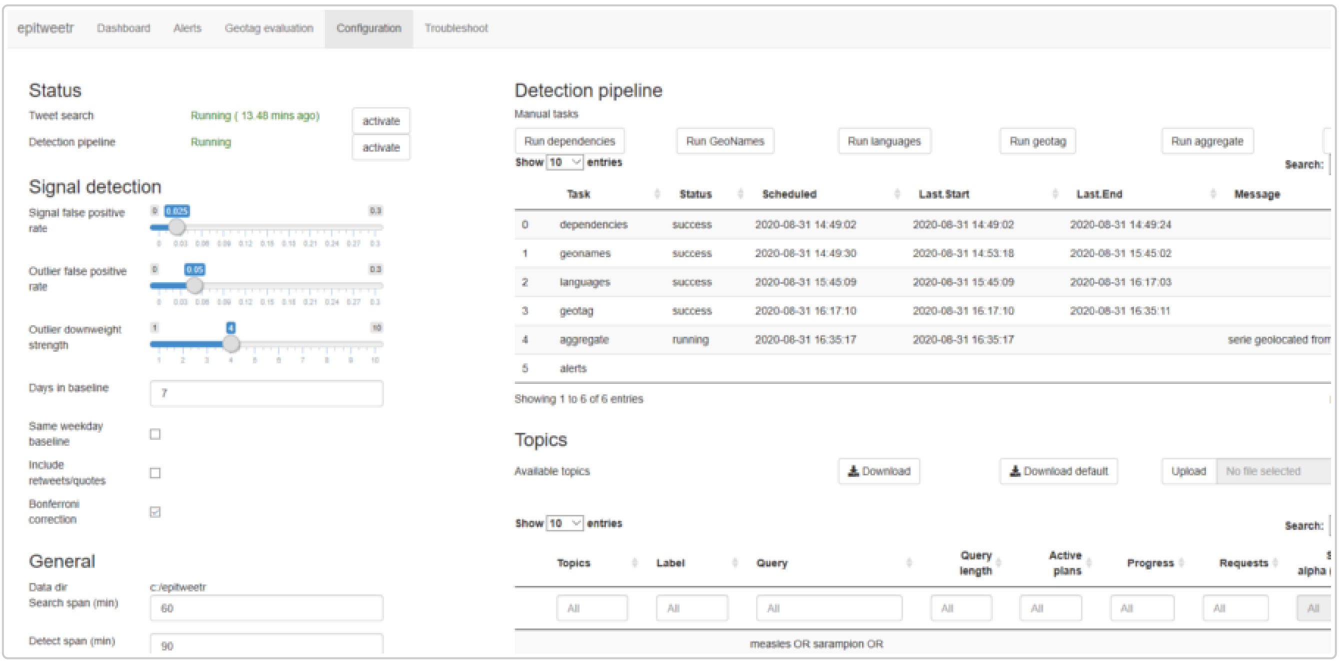
Show 10 entries

Search:

	Tweet ID	Text	Language	Location name	Location type	Country code	Country	Score	Tagged tex
	All	All	All	All	All	All	All	All	All
1	1300400977070247936	RT @PaulaAmaChile: Creo que nunca en mi vida había tenido una mezcla tan grande de sentimientos al ver como un país tan próspero se derrumb...	es	Republic of Chile	PCLI	CL	Republic of Chile	17.878939	Paula,Ama,C
59	1300399708708495360	Joder que rabia me acabo de encontrar un hacker en Sea of Thieves, el tipo se hacia invicible y era invencible. Me... https://t.co/5aGFFZE2GF	es	Republic of Guinea-Bissau	PCLI	GW	Republic of Guinea-Bissau	12.776261	Sea,Thieves
99	1300400913828524037	RT @VitaVirginiaBot: 1 April, 1932 It makes me rage and wake in a hellish misery at dawn. I daresay this kind of outrage is among the real...	en	Republic of Botswana	PCLI	BW	Republic of Botswana	11.979905	Vita,Virginia,I
24	1300400437586165762	@DIEGO_10799 @samgj18 @estebanhop107 @Dani_Matamoros @LujoPaisa Jajajjj men pero si muestras rabia, mas bien relá... https://t.co/qWBSEfWtlX	es	Izúcar de Matamoros	PPLA2	MX	Mexico	11.646905	Matamoros,L
14	1300400635402162176	RT @Golum03477364: Que indignante!!! Entiendo la rabia de Ripoll, soy funcionaria pública y en mi Ministerio pasaba lo mismo, lleno de comp...	es	Ripoll	PPL	ES	Kingdom of Spain	11.0059596	Ripoll
15	1300400632826920960	RT @Golum03477364: Que indignante!!! Entiendo la rabia de Ripoll, soy funcionaria pública y en mi Ministerio pasaba lo mismo, lleno de comp...	es	Ripoll	PPL	ES	Kingdom of Spain	11.0059596	Ripoll
32	1300400307541868544	RT @Golum03477364: Que indignante!!! Entiendo la rabia de Ripoll, soy funcionaria pública y en mi Ministerio pasaba lo mismo, lleno de comp...	es	Ripoll	PPL	ES	Kingdom of Spain	11.0059596	Ripoll

Shiny app configuration page:

Fig 4: Shiny app configuration page



Status

Tweet search: Running (13.48 mins ago)

Detection pipeline: Running

Signal detection

Signal false positive rate: 0.025

Outlier false positive rate: 0.05

Outlier downweight strength: 4

Days in baseline: 7

Same weekday baseline: ☐

Include retweets/quotes: ☐

Bonferroni correction: ☑

General

Data dir: c:/epitweetr

Search span (min): 60

Detect span (min): 90

Detection pipeline

Manual tasks

Show 10 entries

Search:

	Task	Status	Scheduled	Last.Start	Last.End	Message
0	dependencies	success	2020-08-31 14:49:02	2020-08-31 14:49:02	2020-08-31 14:49:24	
1	geonames	success	2020-08-31 14:49:30	2020-08-31 14:53:18	2020-08-31 15:45:02	
2	languages	success	2020-08-31 15:45:09	2020-08-31 15:45:09	2020-08-31 16:17:03	
3	geotag	success	2020-08-31 16:17:10	2020-08-31 16:17:10	2020-08-31 16:35:11	
4	aggregate	running	2020-08-31 16:35:17	2020-08-31 16:35:17		serie geolocated from
5	alerts					

Showing 1 to 6 of 6 entries

Topics

Available topics

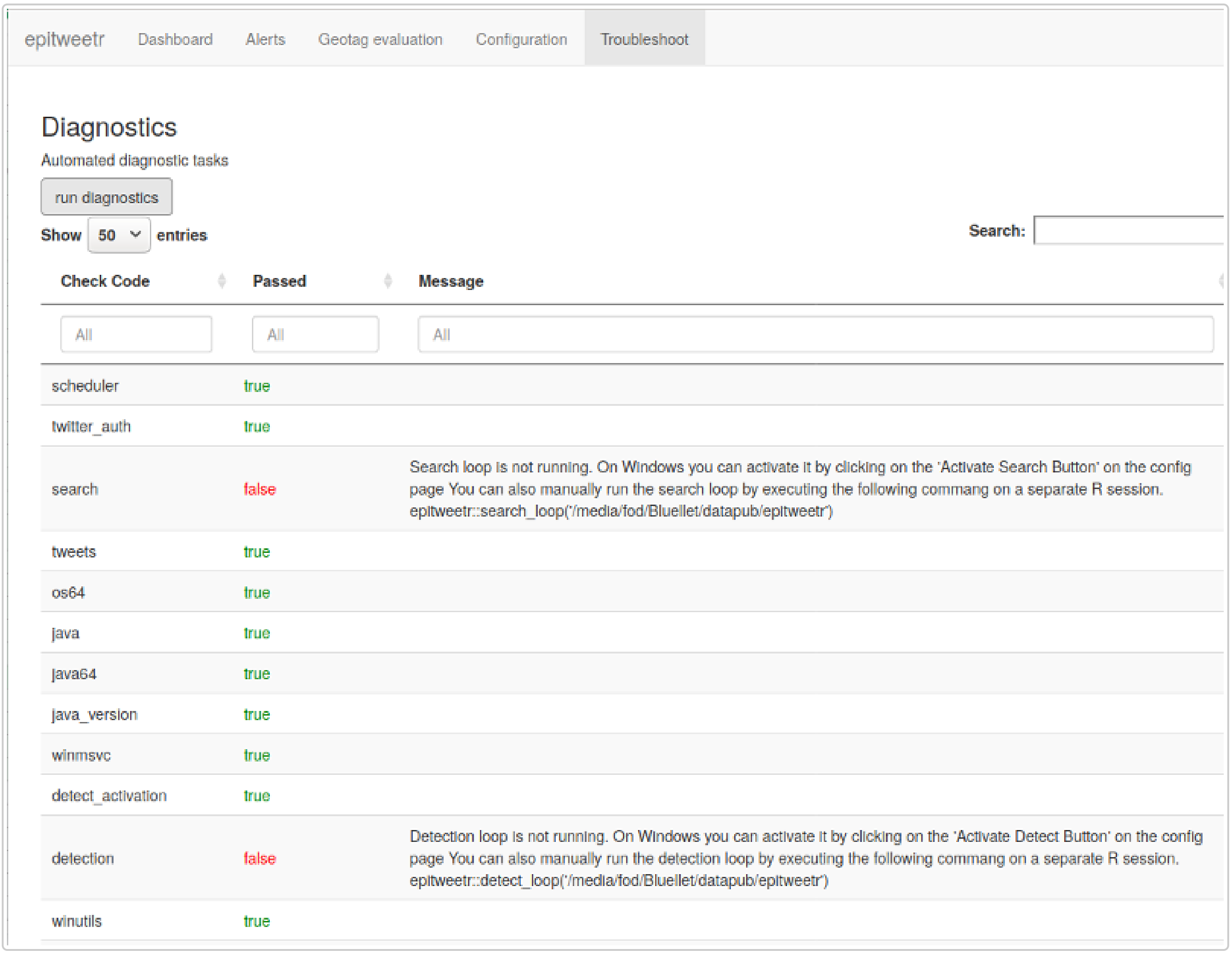
Show 10 entries

Search:

Topics	Label	Query	Query length	Active plans	Progress	Requests	alpha
All	All	All	All	All	All	All	All
		measles OR sarampion OR					

Shiny app troubleshoot page:

Fig 5: Shiny app troubleshoot page



Diagnostics

Automated diagnostic tasks

Check Code	Passed	Message
scheduler	true	
twitter_auth	true	
search	false	Search loop is not running. On Windows you can activate it by clicking on the 'Activate Search Button' on the config page You can also manually run the search loop by executing the following commang on a separate R session. epitweetr::search_loop('/media/fod/Bluellet/datapub/epitweetr')
tweets	true	
os64	true	
java	true	
java64	true	
java_version	true	
winmsvc	true	
detect_activation	true	
detection	false	Detection loop is not running. On Windows you can activate it by clicking on the 'Activate Detect Button' on the config page You can also manually run the detection loop by executing the following commang on a separate R session. epitweetr::detect_loop('/media/fod/Bluellet/datapub/epitweetr')
winutils	true	

Background

Epidemic Intelligence at ECDC

Article 3 of the European Centre for Disease Prevention and Control (ECDC) funding regulation and the Decision No 1082/2013/EU on serious cross-border threats to health have established the detection of public health threats as a core activity of ECDC.

ECDC performs Epidemic Intelligence (EI) activities aiming at rapidly detecting and assessing public health threats, focusing on infectious diseases, to ensure EU's health security. ECDC uses social media as part of its sources to early detect signals of public health threats. Until 2020, the monitoring of social media was mainly performed through the screening and analysis of posts from pre-selected experts or organisations, mainly in Twitter and Facebook.

More information and an online tutorial are available:

EI sources

EI tutorial

Monitoring social media trends

Some signals are not detected at all or are not detected early enough by the methods explained above. The automated monitoring of metadata from social media (e.g. analysis of social media trends) allows the detection of signals that may not be detected by the monitoring of pre-selected accounts in social media, and it improves the timeliness of signal detection.

The analysis of trends from social media by topic, time and place generates relevant signals for early detection.

In 2019, ECDC developed a prototype of a manual R-based tool for the early detection of public health threats from Twitter data. The `epitweetr` package is an extension of this prototype, to allow a wider geolocation of tweets and greater automatisation.

Objectives of epitweetr

The primary objective of `epitweetr` is to use the Twitter Standard Search API version 1.1 in order to detect early signals of potential threats by topic and by geographical unit.

Its secondary objective is to enable the user through an interactive Shiny interface to explore the trend of tweets by time, geographical location and topic, including information on top words and numbers of tweets from trusted users, using charts and tables.

Hardware requirements

The minimum and suggested hardware requirements for the computer are in the table below:

Hardware requirements	Minimum	Suggested
RAM Needed	8GB	16GB recommended
CPU Needed	4 cores	12 cores
Space needed for 3 years of storage	3TB	5TB

The CPU and RAM usage can be configured on the Shiny app configuration page (see section *The interactive user application (Shiny app)>The configuration page*). The RAM, CPU and space needed may depend on the amount and size of the topics you request in the collection process.

Installation

`epitweetr` is conceived to be platform independent, working on Windows, Linux and Mac. We recommend that you use `epitweetr` on a computer that can be run continuously. You can switch the computer off, but you may miss some tweets if the downtime is large enough, which will have implications for the alert detection. Before using `epitweetr`, the following items need to be installed:

Prerequisites for running epitweetr

- R version 3.6.3 or higher
- Java 1.8 eg. openjdk version “1.8” *https://www.java.com/download/*. The 64-bit rather than the 32-bit version is preferred, due to memory limitations. In Mac, also the Java Development Kit *https://docs.oracle.com/javase/9/install/installation-jdk-and-jre-macos.htm*]
- If you are running it in Windows, you will also need Microsoft Visual C++, however in most cases it is likely to be pre-installed:
    - Microsoft Visual C++ 2010 Redistributable Package (x64) *https://www.microsoft.com/en-us/download/details.aspx?id=14632*

Prerequisites for some of the functionalities in epitweetr

- Pandoc, for exporting PDFs and Markdown

    - https://pandoc.org/installing.html
  -
  - Tex installation (TinyTeX or MiKTeX) (or other TeX installation) for exporting PDFs
    - (easiest) https://yihui.org/tinytex/ (install from R, logoff/logon required after installation)
    - https://miktex.org/download (full installation required, logoff/logon required after installation)
  - Machine learning optimisation (only for advanced users)
    - Open Blas (BLAS optimizer), which will speed up some of the geolocation processes: https://www.openblas.net/ Installation instructions: https://github.com/fommil/netlib-Java
    - **or** Intel MKL (https://software.intel.com/content/www/us/en/develop/tools/math-kernel-library/choose-download.html)
  - A scheduler
    - If using Windows, you need to install the R package: taskscheduleR
    - If using Linux, you need to plan the tasks manually
    - If using a Mac, you need to install the R package: cronR

Extra prerequisites for R developers

If you would like to develop `epitweetr` further, then the following development tools are needed:

- Git (source code control) https://git-scm.com/downloads
- Sbt (compiling scala code) https://www.scala-sbt.org/download.html
- If you are using Windows, then you will additionally need Rtools: https://cran.r-project.org/bin/windows/Rtools/

External dependencies

`epitweetr` will need to download some dependencies in order to work. The tool will do this automatically the first time the alert detection process is launched. The Shiny app configuration page will allow you to change the target URLs of these dependencies, which are the following:

- *CRAN JARs*: Transitive dependencies for running Spark, Lucene and embedded scala code. *[https://repo1.maven.org/maven2]*
- *Winutils.exe* (Windows only) This is a Hadoop binary necessary for running SPARK locally on Windows *[http://public-repo-1.hortonworks.com/hdp-win-alpha/winutils.exe]*.

Installing epitweetr from CRAN

After installing all required dependencies listed in the section “Prerequisites for running epitweetr”, you can install `epitweetr`:

```
install.packages(epitweetr)
```

Environment variables

Additionally, the R environment needs to know where the Java installation home is. To check this, type in the R console:

```
Sys.getenv("JAVA_HOME")
```

If the command returns Null or empty, then you will need to set the Java Home environment variable, for your OS, please see your specific OS instructions. In some cases, `epitweetr` can work without setting the Java Home environment variable.

The first time you run the application, if the tool cannot identify a secure password store provided by the operating system, you will see a pop-up window requesting a keyring password (Linux and Mac). This is a password necessary for storing encrypted Twitter credentials. Please choose a strong password and remember it. You will be asked for this password each time you run the tool. You can avoid this by setting a system environment variable named ecdc_twitter_tool_kr_password containing the chosen password.

Launching the epitweetr Shiny app

You can launch the `epitweetr` Shiny app from the R session by typing in the R console. Replace "data_dir" with the designated data directory which is a local folder you choose to store tweets, time series and configuration files in:

```
library(epitweetr)
epitweetr_app("data_dir")
```

Please note that the data directory entered in R should have "/" instead of "" (an example of a correct path would be 'C:/user/name/Documents'). This applies especially in Windows if you copy the path from the File Explorer.

Alternatively, you can use a launcher: In an executable .bat or shell file type the following, (replacing "data_dir" with the designated data directory)

R –vanilla -e epitweetr::epitweetr_app("data_dir")

You can check that all requirements are properly installed in the troubleshoot page. More information is available in section *The interactive user application (Shiny app)>Dashboard:The interactive user interface for visualisation>The troubleshoot page*

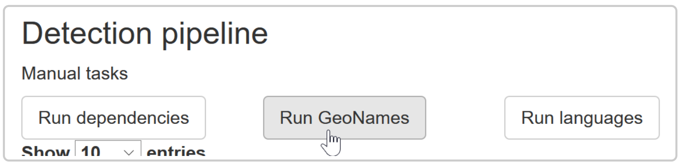
Setting up tweet collection and the alert detection loop

In order to use `epitweetr`, you will need to collect tweets and run the alert detection loop (geonames, languages, geotag, aggregate and alerts). Further details are also available in subsequent sections of the user documentation. A summary of the steps needed is as follows:

- Launch the Shiny app (from the R console)

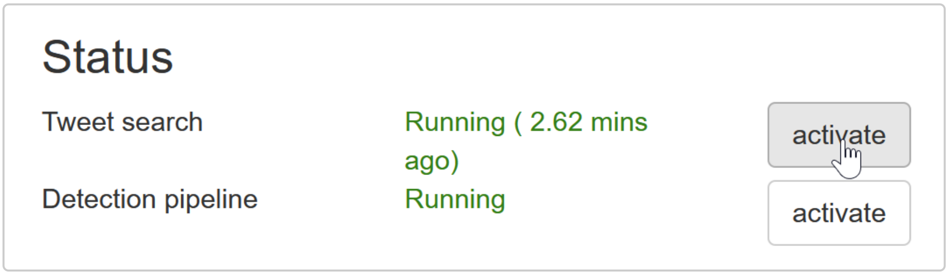
```
library(epitweetr)
epitweetr_app("data_dir")
```

- On the configuration page of the Shiny app, in the manual tasks of the "Detection pipeline", click on "Run dependencies", "Run geonames" and "Run languages" (their status will change to "pending"). This allows the detection pipeline to download the elements needed. As long as no languages are added and no updates are available in geonames.org, these tasks have to be run only the first time you install `epitweetr`.



- Set up the Twitter authentication using a Twitter account or a Twitter developer app, see section *Collection of tweets>Twitter authentication* for more details
- Activate the tweet collection

        - Windows: Click on the “Tweet search” activate button



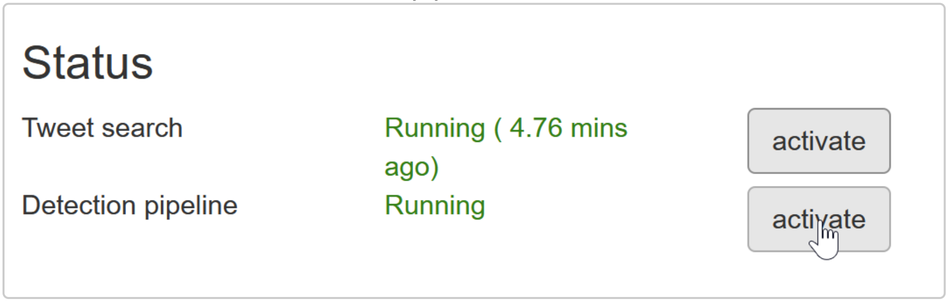
Status

Tweet search	Running (2.62 mins ago)	activate
Detection pipeline	Running	activate

        - Other platforms: In a new R session run the following command

```
library(epitweetr)
search_loop("data_dir")
```

    - You can confirm that the tweet collection is running if the “Tweet search” status is “Running” on the Shiny app configuration page (green text in screenshot above) and “true” in the Shiny app troubleshoot page.
    - Activate the detection pipeline:
        - Windows: Click on the “Detection pipeline” activation button



Status

Tweet search	Running (4.76 mins ago)	activate
Detection pipeline	Running	activate

        - Other platforms: In a new R session run the following command

```
library(epitweetr)
detect_loop("data_dir")
```

    - You can confirm that the detection pipeline is running if the “Detection pipeline” status is “Running” in the Shiny app configuration page and “true” in the Shiny app troubleshoot page.
    - You will be able to visualise tweets after the aggregate step in the detection pipeline table on the Shiny app configuration page is finished and if “Tweet search” is activated.
    - You can start working with the generated signals. **Happy signal detection!**

For more details you can go through the section *How does it work? General architecture behind* `epitweetr`, which describes the underlying processes behind the tweet collection and the signal detection. Also, the section *“The interactive Shiny application (Shiny app)>The configuration page”* describes the different settings on the configuration page.

How does it work? General architecture behind epitweetr

The following sections describe in detail the above general principles. The settings of many of these elements can be configured in the Shiny app configuration page, which is explained in the section *The interactive Shiny application (Shiny app)>The configuration page*.

Collection of tweets

Use of the Twitter Standard Search API version 1.1

`epitweetr` uses the Twitter Standard Search API version 1.1. The advantage of this API is that it is a free service provided by Twitter enabling users of `epitweetr` to access tweets free of charge. The search API is not meant to be an exhaustive source of tweets. It searches against a sample of recent tweets published in the past 7 days and it focuses on relevance and not completeness. This means that some tweets and users may be missing from search results.

While this may be a limitation in other fields of public health or research, the `epitweetr` development team believe that for the objective of signal detection a sample of tweets is sufficient to detect potential threats of importance in combination with other type of sources.

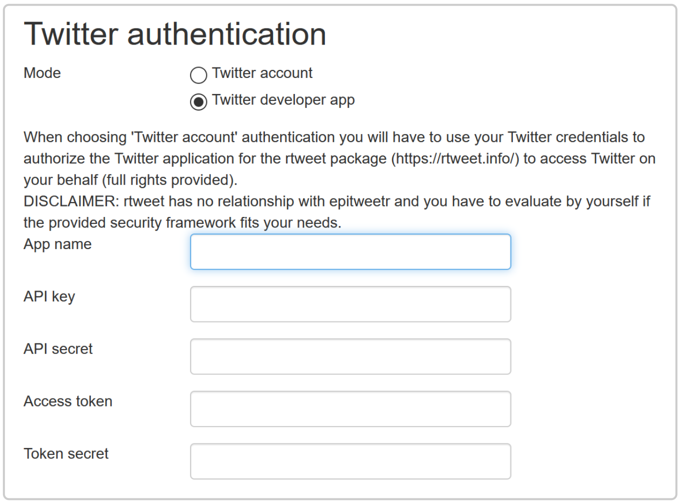
Other attributes of the Twitter Standard Search API version 1.1 include:

- Only tweets from the last 5–8 days are indexed by Twitter
- A maximum of 180 requests every 15 minutes are supported by the Twitter Standard Search API (450 requests every 15 minutes if you are using the Twitter developer app credentials; see next section)
- Each request returns a maximum of 100 tweets and/or retweets

Twitter authentication

You can authenticate the collection of tweets by using a **Twitter account** (this approach utilises the rtweet package app) or by using a **Twitter application**. For the latter, you will need a Twitter developer account, which can take some time to obtain, due to verification procedures. We recommend using a Twitter account via the rtweet package for testing purposes and short-term use, and the Twitter developer application for long-term use.

- **Using a Twitter account:** delegated via rtweet (user authentication)
  - You will need a Twitter account (username and password)
  - The rtweet package will send a request to Twitter, so it can access your Twitter account on your behalf
  - A pop-up window will appear where you can enter your Twitter user name and password to confirm that the application can access Twitter on your behalf. You will send this token each time you access tweets.
- **Using a Twitter developer app:** via `epitweetr` (app authentication)
  - If you have not done so already, you will need to create a Twitter developer account: *https://developer.twitter.com/en/apply-for-access*
  - Create an app
  - For the access type, ensure you have read and write access
  - Make a note of your OAuth settings
    - Add them to the configuration page in the Shiny app (see image below)
    - With this information `epitweetr` can request a token at any time directly to Twitter. The advantage of this method is that the token is not connected to any user information and tweets are returned independently of any user context.
    - With this app you can perform 450 requests every 15 minutes instead of the 180 requests every 15 minutes that a Twitter account allows.


Twitter authentication

Mode	Twitter account [✓ Twitter developer app]

When choosing 'Twitter account' authentication you will have to use your Twitter credentials to authorize the Twitter application for the rtweet package (https://rtweet.info/) to access Twitter on your behalf (full rights provided).
DISCLAIMER: rtweet has no relationship with epitweetr and you have to evaluate by yourself if the provided security framework fits your needs.

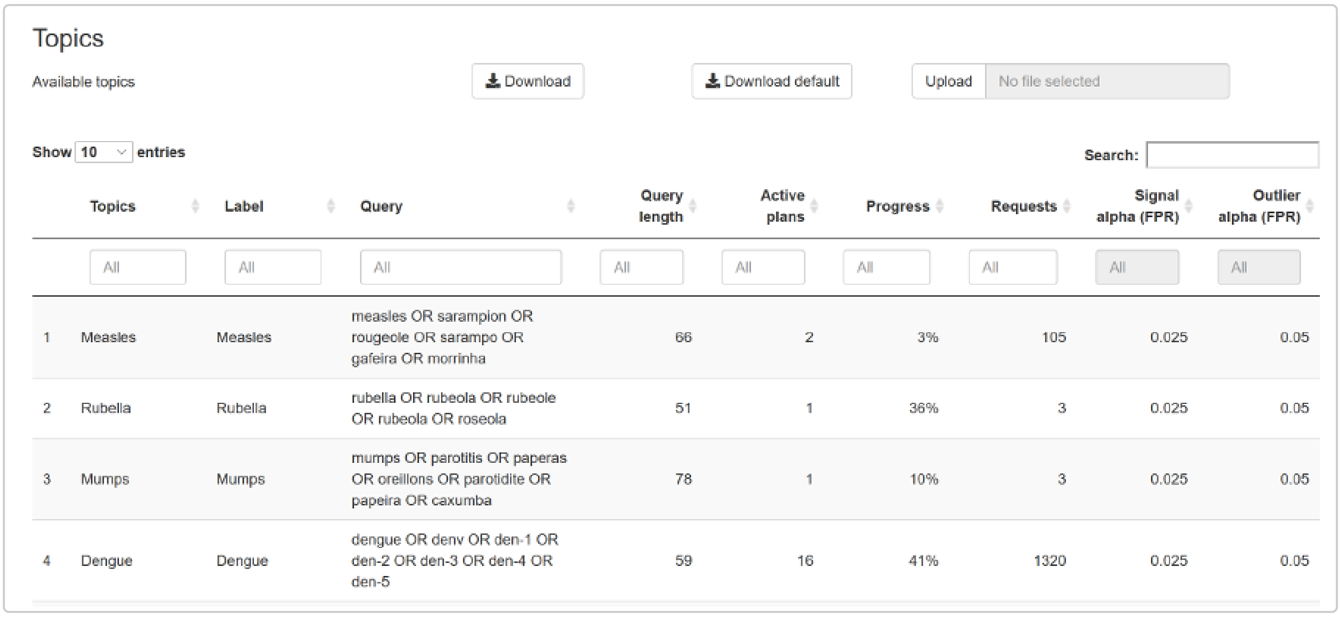
App name	
API key	
API secret	
Access token	
Token secret	

Topics and tweet collection queries

After the Twitter authentication, you need to specify a list of topics in `epitweetr` to indicate which tweets to collect. For each topic, you have one or more queries that `epitweetr` uses to collect the relevant tweets (e.g. several queries for a topic using different terminology and/or languages).

A query consists of keywords and operators that are used to match tweet attributes. Keywords separated by a space indicate an AND clause. You can also use an OR operator. A minus sign before the keyword (with no space between the sign and the keyword) indicates the keyword should not be in the tweet attributes. While queries can be up to 512 characters long, best practice is to limit your query to 10 keywords and operators and limit complexity of the query, meaning that sometimes you need more than one query per topic.

`epitweetr` comes with a default list of topics as used by the ECDC Epidemic Intelligence team at the date of package generation (1st of September, 2020). You can view details of the list of topics in the Shiny app configuration page (see screenshot below).


Topics

Available topics

	Topics	Label	Query	Query length	Active plans	Progress	Requests	Signal alpha (FPR)	Outlier alpha (FPR)
	All	All	All	All	All	All	All	All	All
1	Measles	Measles	measles OR sarampion OR rougeole OR sarampo OR gafeira OR morrinha	66	2	3%	105	0.025	0.05
2	Rubella	Rubella	rubella OR rubeola OR rubeole OR rubeola OR roseola	51	1	36%	3	0.025	0.05
3	Mumps	Mumps	mumps OR parotitis OR paperas OR oreillons OR parotidite OR papeira OR caxumba	78	1	10%	3	0.025	0.05
4	Dengue	Dengue	dengue OR denv OR den-1 OR den-2 OR den-3 OR den-4 OR den-5	59	16	41%	1320	0.025	0.05

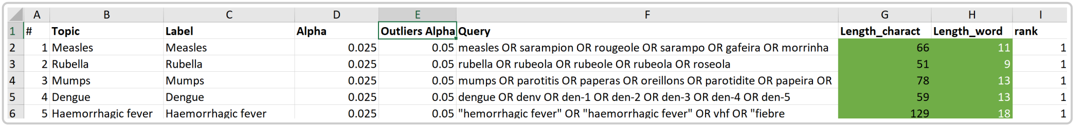
On the configuration page, you can also download the list of topics, modify it and upload it to `epitweetr`. The new list of topics will then be used for tweet collection and visible in the Shiny app. The list of topics is an Excel file (*.xlsx) as it handles user-specific regional settings (e.g. delimiters) and special characters well. You can create your own list of topics and upload it too, noting that the structure should include at least:

- The name of the topic, with the header “Topic” in the Excel spreadsheet. This name should include alphanumeric characters, spaces, dashes and underscores only. Note that it should start with a letter.

- The query, with the header “Query” in the Excel spreadsheet. This is the query `epitweetr` uses in its requests to obtain tweets from the Twitter Standard Search API. See above for syntax and constraints of queries.

The topics.xlsx file additionally includes the following fields:

- An ID, with the header “#” in the Excel spreadsheet, noting a running integer identifier for the topic.
- A label, with the header “Label” in the Excel spreadsheet, which is what is displayed in the drop-down topic menu of the Shiny app tabs.
- An alpha parameter, with the header “Signal alpha (FPR)” in the Excel spreadsheet. FPR stands for “false positive rate”. Increasing the alpha will decrease the threshold for signal detection, resulting in an increased sensitivity and possibly obtaining more signals. Setting this alpha can be done empirically and according to the importance and nature of the topic.
- “Length_charact” is an automatically generated field that calculates the length of all characters used in the query. This field is helpful as a request should not exceed 500 characters.
- “Length_word” indicates the number of words used in a request, including operators. Best practice is to limit your number of keywords to 10.
- An alpha parameter, with the header “Outlier alpha (FPR)” in the Excel spreadsheet. FPR stands for “false positive rate”. This alpha sets the false positive rate for determining what an outlier is when downweighting previous outliers/signals. The lower the value, the fewer previous outliers will potentially be included. A higher value will potentially include more previous outliers.
- “Rank” is the number of queries per topic



	A	B	C	D	E	F	G	H	I
1	#	Topic	Label	Alpha	Outliers Alpha	Query	Length_charact	Length_word	rank
2	1	Measles	Measles	0.025	0.05	measles OR sarampion OR rougeole OR sarampo OR gafeira OR morrinha	66	11	1
3	2	Rubella	Rubella	0.025	0.05	rubella OR rubeola OR rubeole OR rubeola OR roseola	51	9	1
4	3	Mumps	Mumps	0.025	0.05	mumps OR parotitis OR paperas OR oreillons OR parotidite OR papeira OR	78	13	1
5	4	Dengue	Dengue	0.025	0.05	dengue OR denv OR den-1 OR den-2 OR den-3 OR den-4 OR den-5	59	13	1
6	5	Haemorrhagic fever	Haemorrhagic fever	0.025	0.05	"hemorrhagic fever" OR "haemorrhagic fever" OR vhf OR "fiebre	129	18	1

When uploading your own file, please modify the topic and query fields, but do not modify the column titles.

Scheduled plans to collect tweets

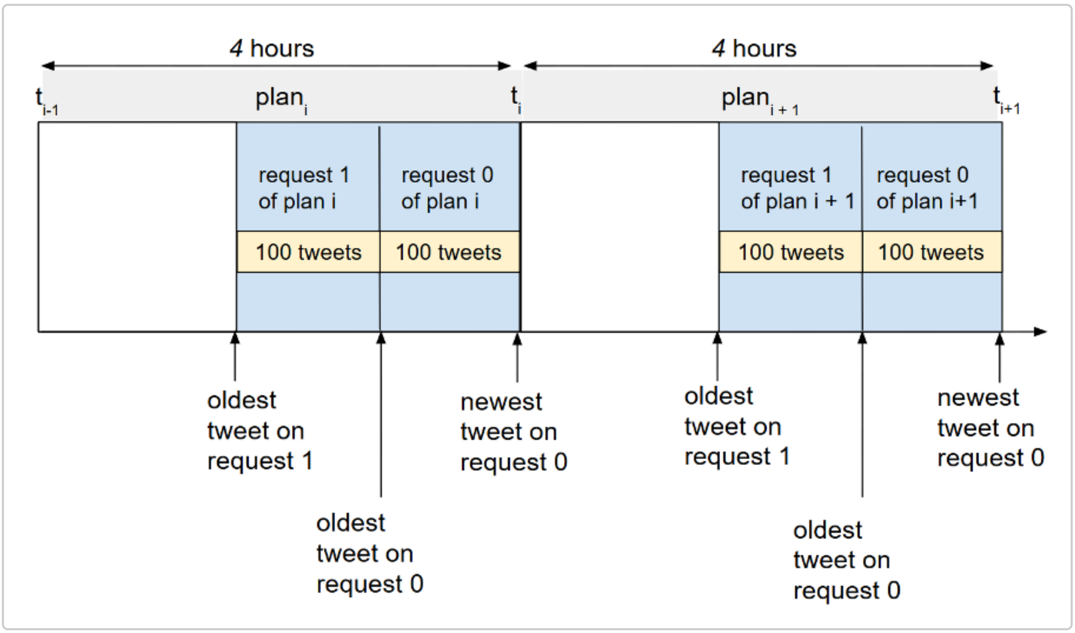
As a reminder, `epitweetr` is scheduled to make 180 requests (queries) to Twitter every 15 minutes (or 450 requests every 15 minutes if you are using Twitter developer app credentials). Each request can return 100 tweets. The requests return tweets and retweets. These are returned in JSON format, which is a light-weight data format.

In order to collect the maximum number of tweets, given the Standard Search API limitations, and in order for popular topics not to prevent other topics from being adequately collected, `epitweetr` uses “search plans” for each query.

The first “search plan” for a query will collect tweets from the current date-time backwards until 7 days (7 days because of the Standard Search API limitation) before the current “search plan” was implemented. The first “search plan” is the biggest, as no tweets have been collected so far.

All subsequent “search plans” are scheduled intervals that are set up in the configuration page of the `epitweetr` Shiny app (see section *The interactive Shiny app > the configuration page > General*). For illustration purposes, let us consider the search plans are scheduled at four-hour intervals. The plans collect tweets for a specific query from the current date-time back until four hours before the date-time when the current “search plan” is implemented (see image below). `epitweetr` will make as many requests (each returning up to 100 tweets) during the four-hour interval as needed to obtain all tweets created within that four-hour interval.

For example, if the “search plan” begins at 4 am on the 10th of September 2020, `epitweetr` will launch requests for tweets corresponding to its queries for the four-hour period from 4 am to midnight on the 10th of September 2020. `epitweetr` starts by collecting the most recent tweets (the ones from 4 am) and continues backwards. If during the four-hour time period between 4am and midnight the API does not return any more results, the “search plan” for this query is considered completed.



However, if topics are very popular (e.g. COVID-19 in 2020), then the “search plan” for a query in a given four-hour window may not be completed. If this happens, `epitweetr` will move on to the “search plans” for the subsequent four-hour window, and put any previous incomplete “search plans” in a queue to execute when “search plans” for this new four-hour window are completed.

Each “search plan” stores the following information:

Field	Type	Description
expected_end	Timestamp	End DateTime of the current search window
scheduled_for	Timestamp	The scheduled DateTime for the next request. On plan creation this will be the current DateTime and after each request this value will be set to a future DateTime. To establish the future DateTime, the application will estimate the number of requests necessary to finish. If it estimates that N requests are necessary, the next schedule will be in 1/N of the remaining time.
start_on	Timestamp	The DateTime when the first request of the plan was finished
end_on	Timestamp	The DateTime when the last request of the plan was finished if that request reached a 100% plan progress.
max_id	Long	The max Twitter id targeted by this plan, which will be defined after the first request
since_id	Long	The last tweet id returned by the last request of this plan. The next request will start collecting tweets before this value. This value is updated after each requests and allows the Twitter API to return tweets before min_time(p_i)
since_target	Long	If a previous plan exists, this value stores the first tweet id that was downloaded for that plan. The current plan will not collect tweets before that id. This value allows the Twitter API to return tweets after p_i-time_back
requests	Int	Number of requests performed as part of the plan
progress	Double	Progress of the current plan as a percentage. It is calculated as (current$max_id - current$since_id)/(current$max_id - current$since_target). If the Twitter API returns no tweets the progress is set to 100%. This only applies for non error responses containing an empty list of tweets.

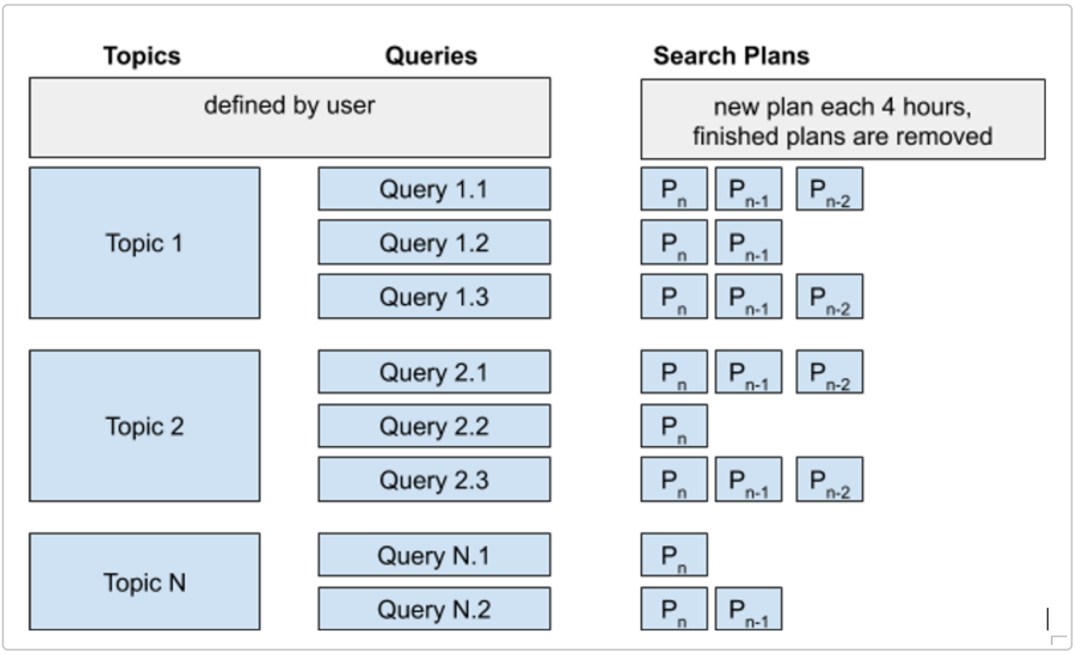
`epitweetr` will execute plans according to these rules:

- `epitweetr` will detect the newest unfinished plan for each search query with the scheduled_for variable located in the past.
- `epitweetr` will execute the plans with the minimum number of requests already performed. This ensures that all scheduled plans perform the same number of requests.
- As a result of the two previous rules, requests for topics under the 180 limit of the Twitter Standard Search API (or 450 if you are using Twitter developer app authentication) will be executed first and will produce higher progress than topics over the limit.

The rationale behind this is that topics with such a large number of tweets that the 4-hour search window is not sufficient to collect them, are likely to already be a known topic of interest. Therefore, priority should be given to smaller topics and possibly less well-known topics.

An example is the COVID-19 pandemic in 2020. In early 2020, there was limited information available regarding COVID-19, which allowed detecting signals with meaningful information or updates (e.g. new countries reporting cases or confirming that it was caused by a coronavirus). However, throughout the pandemic, this topic became more popular and the broad topic of COVID-19 was not effective for signal detection and was taking up a lot of time and requests for `epitweetr`. In such a case it is more relevant to prioritise the collection of smaller topics such as sub-topics related to COVID-19 (e.g. vaccine AND COVID-19), or to make sure you do not miss other events with less social media attention.

If search plans cannot be finished, several search plans per query may be in a queue:



Geolocation

In a parallel process to the collection of tweets, `epitweetr` attempts to geolocate all collected tweets using an unsupervised machine learning process. This process runs in the schedule window defined by “Detect span” property in the configuration page under “General settings” (e.g. if a four-hour window is set it will run every four hours and will geolocate all tweets collected since the last time it successfully ran).

`epitweetr` stores two types of geolocation for a tweet: tweet location, which is geolocation information within the text of a tweet (or a retweeted or quoted tweet), and user location from the available metadata. For signal detection, the best location is used while in the dashboard both types can be visualised.

Geolocation based on tweet location

The tweet location is extracted and stored by `epitweetr` based on the geolocation information found within a tweet text. In case of a retweet or quoted tweet, it will extract the geolocation information from the original tweet text that was retweeted or quoted. If neither is available, no tweet location is stored based on tweet text.

`epitweetr` identifies if a tweet text contains reference to a particular location by breaking down the tweet text into sets of words and evaluating those which are more likely to be a location by using a machine learning model. The algorithm also adds further words (one by one) to the set and if the score grows by using more words, the algorithm tries to find a local maximum of the score with a bigger text. It then matches these words against a reference database, which is geonames.org. This is a geographical database available and accessible through various web services, under a Creative Commons attribution license. The GeoNames.org database contains over 25,000,000 geographical names. `epitweetr` uses by default those limited to currently existing ones and those with a known population (so just over 500,000 names). You can override this default on the Shiny app configuration page, by unchecking “Simplified geonames”. The database also contains longitude and latitude attributes of localities and variant spellings (cross-references), which are useful for finding purposes, as well as non-Roman script spellings of many of these names.

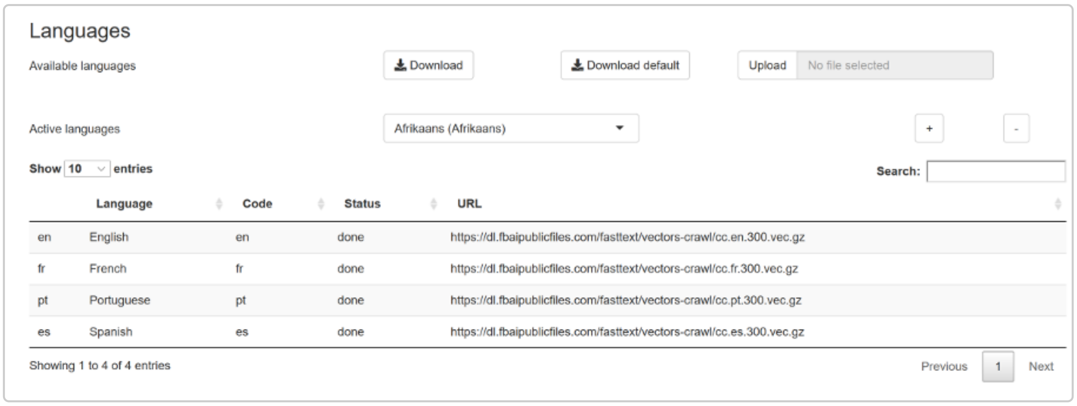
The matches can be performed at any level of administrative hierarchy. The matching is powered by Apache Lucene, which is an open-source high-performance full-featured text search engine library.

Some text may have several locations through the matching process. However, only the location with the highest score will be selected.

A higher score is associated with a greater probability that a match is correct. A score is:

- Higher if unusual parts of the name are matched
- Higher if several administrative levels are matched
- Higher if location population is bigger
- Higher for countries and cities vs administrative levels
- Higher for capital letter acronyms like NY
- Lower for words that are more likely to be other (non-geographical) kinds of words. For example “Fair Play” town in Colorado. This is achieved by using language models provided by fasttext.cc.

You can select which languages you would like to check for other kinds of words in, by selecting the active language desired within the configuration page of the Shiny app and clicking on the “+” icon:



Active languages: Afrikaans (Afrikaans)

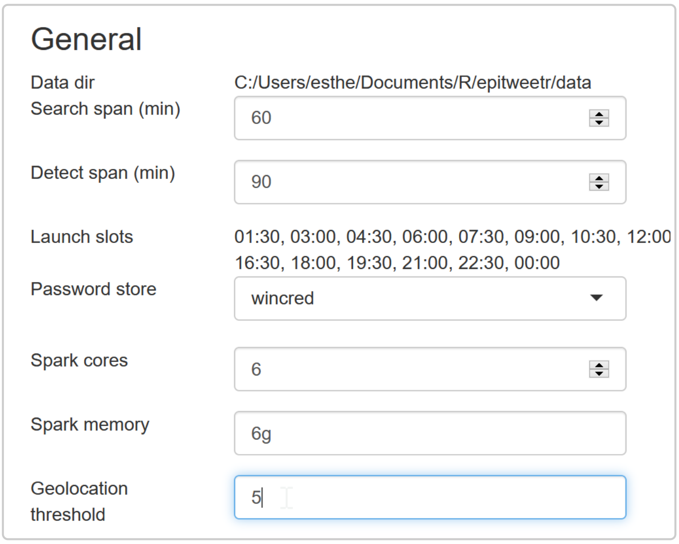
	Language	Code	Status	URL
en	English	en	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.en.300.vec.gz
fr	French	fr	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.fr.300.vec.gz
pt	Portuguese	pt	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.pt.300.vec.gz
es	Spanish	es	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.es.300.vec.gz

Showing 1 to 4 of 4 entries

In addition, you can unselect languages by selecting the language within the configuration page of the Shiny app and clicking on the “-” icon: ADD IMAGE HERE

A minimum score (the “geolocation threshold”) can be globally set in the general settings on the configuration page to reduce the number of false positives (see image). All geolocations with a smaller score than the geolocation threshold will be discarded by the algorithm as tweet location. If there is more than one match over the minimum score, then the match with the highest score will be chosen.

The threshold is empirically chosen and can be evaluated against a human read of tweets and tweet locations, on the geotag evaluation page.


General

Data dir C:/Users/esthe/Documents/R/epitweetr/data

Search span (min) 60

Detect span (min) 90

Launch slots 01:30, 03:00, 04:30, 06:00, 07:30, 09:00, 10:30, 12:00
16:30, 18:00, 19:30, 21:00, 22:30, 00:00

Password store wincred

Spark cores 6

Spark memory 6g

Geolocation threshold 5

Geolocation based on user location

Different types of user locations are available from the metadata provided through the Twitter Standard Search API. epitweetr selects the best user location for the aggregated files using the following order:

- the user's exact or approximate location at time of the tweet (provided by the API)
- if the user's location is not available and the tweet is a retweet or a quoted tweet, then the user's exact or approximate location at time of the retweet/quoted tweet is used (provided by the API)
- if not available, then the user declared location is used
- if not available, then the "home" in the public profile is used.

With exact locations, the longitude and latitude are provided. If it is the estimated location, `epitweetr` calculates the longitude and latitude from GeoNames.org.

If user location information provided by the API is not available, then `epitweetr` will calculate longitude and latitude from the user declared location or the place name given in the user "public profile", using GeoNames.org.

Stored geolocated tweet information

The geolocation of the match is stored as a country code (using the ISO 3166 standard) and a longitude and latitude associated with the exact geolocation in the aggregated data.

Most frequent words found in tweets

As the number of tweets and words can be very large, tweets are analysed for most frequent words in chunks of 10 000 tweets to get the top 500 words of each chunk for the same language, day and topic.

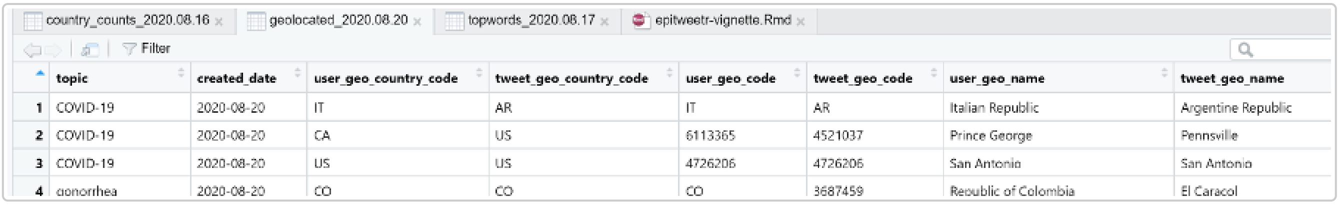
To ensure reasonable performance of `epitweetr`, the 500 top words are first determined globally, and then these words are used to subset by country, to get the top words by country, day and topic. In very small locations with few number of tweets geolocated, there may be no frequent words available.

Note that, differently to the other visualisation figures, the most frequent words in tweets are always based on the geolocation related to "tweet location" and not to "user location" regardless of the filter selected in the dashboard.

Aggregation of data

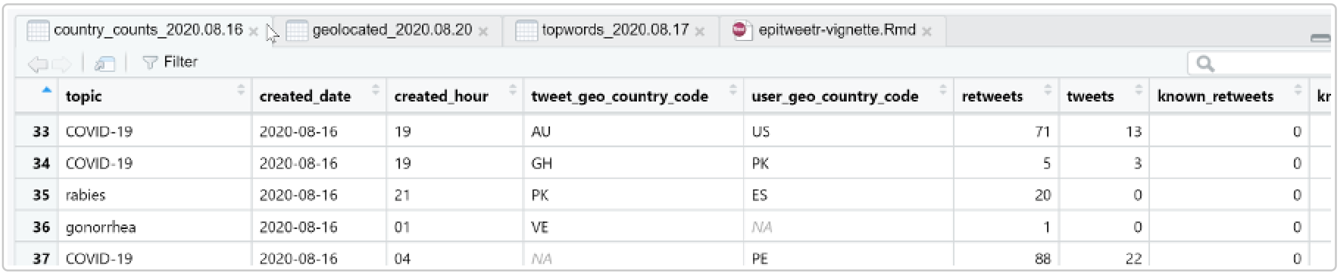
The aggregation process produces three Rds files (a native R format): geolocated, country_counts and topwords.

In the geolocated Rds file, the number of tweets or retweets are stored by topic, date, tweet text geolocation longitude and latitude and user geolocation longitude and latitude. Each of these entries also has the country associated with the tweet text geolocation and the country associated with the user geolocation (see partial screenshot below). Note that tweets without geolocation information are also included.



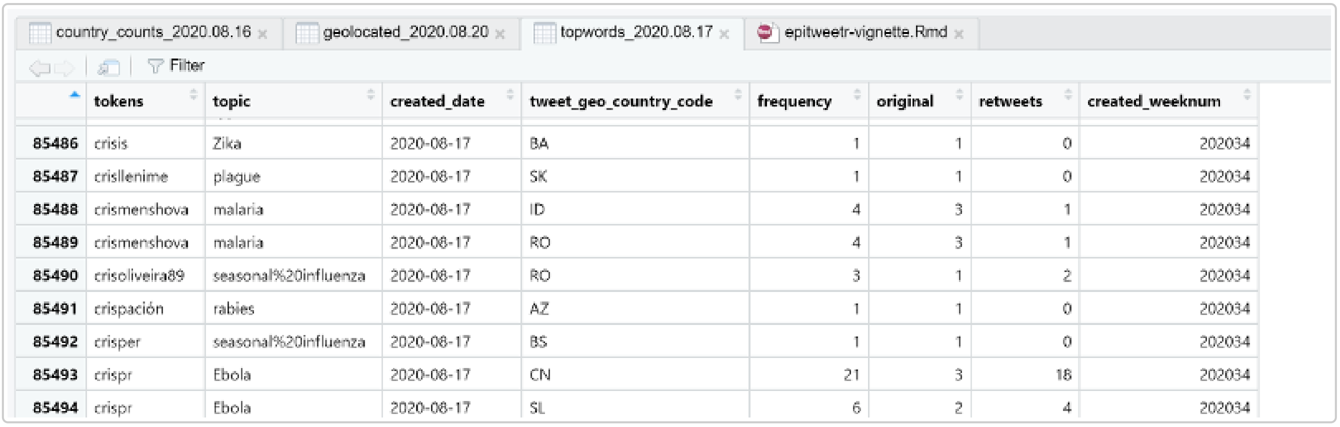
	topic	created_date	user_geo_country_code	tweet_geo_country_code	user_geo_code	tweet_geo_code	user_geo_name	tweet_geo_name
1	COVID-19	2020-08-20	IT	AR	IT	AR	Italian Republic	Argentine Republic
2	COVID-19	2020-08-20	CA	US	6113365	4521037	Prince George	Pennsville
3	COVID-19	2020-08-20	US	US	4726206	4726206	San Antonio	San Antonio
4	gonorrhea	2020-08-20	CO	CO	CO	3687459	Republic of Colombia	El Caracol

The country_counts Rds file is used to create the trendline in the Shiny app. This is a smaller Rds file, without the longitude and latitude information, and includes the number of tweets by hour within a day, by country (according to tweet location or user location), topic (see screenshot), and whether a tweet was a retweet or not. The known_retweets and known_original fields give the number of tweets or retweets from a list of “important users”. In this file, tweets without geolocation are also included. Including tweets without geolocation information enables you to view all tweets when selecting “world” as a region, regardless if geolocation was successful or not.



	topic	created_date	created_hour	tweet_geo_country_code	user_geo_country_code	retweets	tweets	known_retweets	kr
33	COVID-19	2020-08-16	19	AU	US	71	13	0	
34	COVID-19	2020-08-16	19	GH	PK	5	3	0	
35	rabies	2020-08-16	21	PK	ES	20	0	0	
36	gonorrhea	2020-08-16	01	VE	NA	1	0	0	
37	COVID-19	2020-08-16	04	NA	PE	88	22	0	

The aggregation by top words is stored in the topwords.Rds file, and displays the number of tweets or retweets (or both) by topic, top word, date, country of tweet location and whether a tweet was a retweet or not (see screenshot).



	tokens	topic	created_date	tweet_geo_country_code	frequency	original	retweets	created_weeknum
85486	crisis	Zika	2020-08-17	BA	1	1	0	202034
85487	crisllenime	plague	2020-08-17	SK	1	1	0	202034
85488	crismenshova	malaria	2020-08-17	ID	4	3	1	202034
85489	crismenshova	malaria	2020-08-17	RO	4	3	1	202034
85490	crisoliveira89	seasonal%20influenza	2020-08-17	RO	3	1	2	202034
85491	crispación	rabies	2020-08-17	AZ	1	1	0	202034
85492	crisper	seasonal%20influenza	2020-08-17	BS	1	1	0	202034
85493	crispr	Ebola	2020-08-17	CN	21	3	18	202034
85494	crispr	Ebola	2020-08-17	SL	6	2	4	202034

Signal detection

The main objective of `epitweetr` is to detect signals in the observed data streams, i.e. counts in the aggregated time series that exceed what is expected. For detecting signals, `epitweetr` uses an extended version of the EARS (Early Aberration Reporting System) algorithm (Fricker, Hegler, and Dunfee 2008), which in what follows is denoted by `eears` (extended EARS). This algorithm is part of the R package `surveillance` (Salmon, Schumacher, and Höhle 2016).

As a default it uses a moving window of the past seven days to calculate a threshold. If the count for the current day exceeds this threshold, then a signal is generated.

Details of the algorithm underlying signal detection

The `eears` algorithm is applied on the counts from the past seven 24-hour blocks prior to the current 24 hour block of the signal detection. The running mean and the running standard deviation are calculated:

$$\overline{y}_0 = \frac{1}{7}\sum_{t=-7}^{-1} y_t \quad \text{and} \quad s_0^2 = \frac{1}{7-1}\sum_{t=-7}^{-1}(y_t - \overline{y}_0)^2,$$

where $y_t, t = \ldots, -2, -1, 0$ denotes the observed count data time series with time index 0 denoting the current block. Furthermore, the time index $-7, \ldots, -1$ denote the seven blocks prior to the current block.

Under the null hypothesis of no spikes, it is assumed that the y_t are identically and independently $N(\mu, \sigma^2)$ distributed with unknown mean μ and unknown variance σ^2. Hence, the upper limit of a simple one-sided $(1-\alpha)\times$ 100% plug-in prediction interval for y_0 based on $y_{-7}, \ldots, y_{-1}$ is given as

$$U_0 = \overline{y}_0 + z_{1-a} \times s_0,$$

where z_{1-a} is the (1 − α)- quantile of the standard normal distribution. An alert is raised if $y_0 > U_0$. If one uses α=0.025, then this corresponds to investigating, if y_0 exceeds the estimate for the mean plus 1.96 times the standard deviation. However, as pointed out by Allévius and Höhle (2017), the correct approach would be to compare the observation to the upper limit of a two-sided 95% prediction interval for y_0, because this respects both the sampling variation of a new observation *and* the uncertainty originating from the parameter estimation of the mean and variance. Hence, the statistical appropriate form is to compute the upper limit by

$$U_0 = \overline{y_0} + t_{1-a}(7-1) \times s_0 \times \sqrt{1+\frac{1}{7}}.$$

where $t_{1-a}(k-1)$ denotes the 1 − α quantile of the t-distribution with k − 1 degrees of freedom.

Downweighting previous signals

If previous signals are included without modification in the historic values when calculating the running mean and standard deviation for the signal detection, then the estimated mean and standard deviation might become too large. This may mean that important current signals will not be detected. To address this issue, `epitweetr` downweights previous signals, such that the mean and standard deviation estimation is adjusted for such outliers using an approach similar to that used in the C. P. Farrington et al. (1996). Historic values that are not identified as previous signals are given a weight of “1”. Similarly, historic values identified as signals are given a weight lower than one and a new fit is performed using these weights (scaled s.t. they again sum to 7 observations). Details on the downweighting procedure can be found in Annex I of this user documentation.

Timing of signal detection

Signal detection is carried out based on “days”, which are moving windows of 24 hours, moving according to the detect span (see also section *The interactive user application (Shiny app) > The configuration page > General*). The baseline is calculated on these “days” from -1 to -8 (if the current “day” is zero).

Signals are generated according to the detect span (see section *The interactive user application (Shiny app) > The configuration page > General*), with

- general email alerts either following this detect span (e.g. if the detect span was four hours, the email alerts will be sent every four hours)
- email alerts in real-time. In these email alerts, previously generated signals will be omitted.

The different types of email alerts for each user can be specified in the configuration page (see section *The interactive user application (Shiny app) > The configuration page > General*).

The alpha parameter: the false positive rate of the signal detection

A key attribute of signal detection is the ability of an algorithm to detect true threats or events without overloading the investigators with too many false positives. In this way, the alpha parameter determines the threshold of the detection interval. If the alpha is high, then more potential signals are generated and if the alpha is low fewer potential signals are generated (but potential threats or events could be missed). The setting of the alpha is often done empirically, and depends also on the resources of those investigating the signals and the importance of missing a potential threat or event.

There is a global alpha, that can be set/changed in the `epitweetr` configuration page under “Signal false positive rate” (see section *The interactive user application (Shiny app)* > *The configuration page* > *General*). Additionally, the default alpha can be overridden in the topics list. Here, if you like, you can associate each topic with a specific alpha, depending on the estimated public health importance of the topic or potential associated event or threat.

Bonferroni correction

To account for multiple testing, for country-specific signal detection, as a default, the alpha is divided by the number of countries. For continent-specific signal detection, the alpha is divided by the number continents. This is a Bonferroni correction for multiple testing.

To override this, you can uncheck “Bonferroni correction” in the “Signal detection” part of the configuration page in the Shiny app.

Using same weekdays as baseline

It is possible that there is a “day of the week effect”, where more tweets may be tweeted on a given day of the week (e.g. Monday) than on other days. To avoid this, you can also choose to calculate the baseline not on consecutive days, but on the past N days that correspond to the same 24 hour window N days back. This way if N = 7, the baseline is calculated using the “days” from -7, -14, -21, -28, -35, -42, -49 and -56 (if the current “day” is zero).

This option is on the configuration page of the Shiny app “Default same weekday baseline”.

Sending email alerts

Emails containing a list of signals detected are sent automatically by `epitweetr` according to the detect span and the subscribers list. Due to the time necessary to collect, geolocate and aggregate the tweets, email alerts will miss the most recent tweets that have not yet gone to these processes. The lag between tweets and alerts is expected to be less than (2 * (collect_span) + detect_span) which should be 3h30 using default values.

The email alerts will include the following information on the signals for each topic:

- The date and hour the signal was detected
- The geographical location(s) where the signal was detected
- The most frequent words (top words) in the tweets
- The number of tweets and the threshold
- The percentage of tweets from important users
- Information on the settings, such as: was the Bonferroni correction used, was the same weekday baseline used, were retweets included, etc.

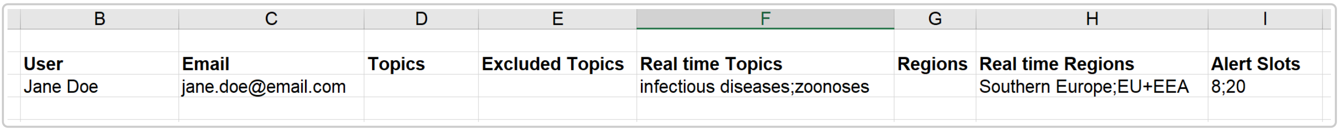
This information is also available in the alerts page of the Shiny app.

The subscribers can receive real-time alerts (i.e. as soon as the detection loop is finalised) or scheduled alerts (e.g. once or twice a day). The subscribers list can be changed in the configuration page by downloading the Excel spreadsheet. This file has the following variables:

- “User”: name of the subscriber (e.g. Jane Doe).
- “Email”: email of the subscriber (e.g. jane.doe@email.com).

- “Topics”: list of topics for which the subscriber will receive scheduled alerts. The names used must match the column “Topic” in the list of topics.
- “Excluded”: topic for which the subscribers will not receive scheduled alerts.
- “Real time Topics”: list of topics for which the subscriber will receive real-time alerts.
- “Regions”: list of regions for which the subscriber will receive scheduled alerts.
- “Real time Regions”: list of regions for which the subscriber will receive real-time alerts.
- “Alert Slots”: these are the detection loop slots after which the subscriber will receive the scheduled alert. Available slots can be taken from “Launch slots” in the “General” section of the configuration page. If no value is included, the subscriber will receive real-time alerts for all topics and regions, even if there are real-time topics or regions specified in the Excel spreadsheet.

When including more than one topic and/or region in the subscribers list, these should be separated by semi-colon (;) with no spaces (e.g. Ebola;infectious diseases;dengue). The names must match the column “Topics” in the list of topics and the column “Name” in the contry/region list from the configuration page.



B	C	D	E	F	G	H	I
User	**Email**	**Topics**	**Excluded Topics**	**Real time Topics**	**Regions**	**Real time Regions**	**Alert Slots**
Jane Doe	jane.doe@email.com			infectious diseases;zoonoses		Southern Europe;EU+EEA	8;20

Folder structure

`epitweetr` stores tweets, aggregated tweets and the configuration in the “data folder” you have to designate when launching the application.

Within the **data** folder there are 3 JSON files:

- properties.json, generated from the information from the General properties of the Shiny app
- topics.json managed by the search loop: it keeps a track of tweet collection plans and progress
- tasks.json managed by the detect loop: it keeps information and status of the different tasks done by this process.

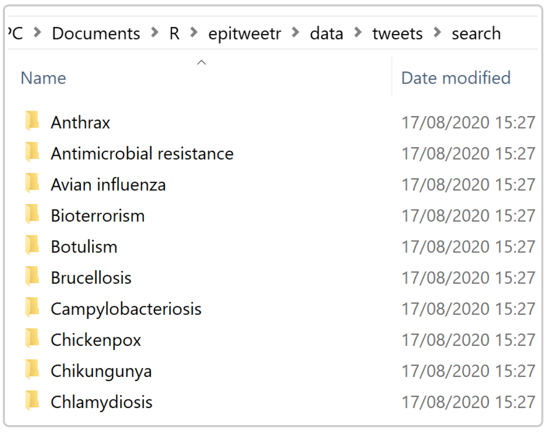
There are also the following subfolders:

- “geo”, which stores the GeoNames data as text and index files
- “hadoop”, which stores Spark dependencies for Windows operating systems
- “jars”, where collections of Java dependencies are stored that are needed in the geolocation and aggregate processes
- “languages”, where the fasttext files indexes and models are stored that are used to perform geolocation in the tweet text
- “stats”, where json files are stored reporting statistics used to optimise the aggregate process by linking tweet files and posted dates of tweets
- “alerts” that store json files of alerts detected by the detect loop.
- “tweets” and “series” that are explained in more detail below.

Data folder > tweets

In the data folder, the subfolder “tweets” has two further subfolders: **search** and **geolocated**

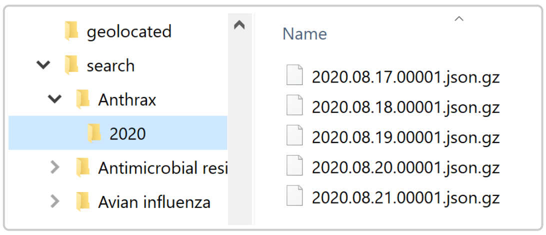
The *search folder* contains subfolders for each topic listed in the topics list:



C › Documents › R › epitweetr › data › tweets › search

Name	Date modified
Anthrax	17/08/2020 15:27
Antimicrobial resistance	17/08/2020 15:27
Avian influenza	17/08/2020 15:27
Bioterrorism	17/08/2020 15:27
Botulism	17/08/2020 15:27
Brucellosis	17/08/2020 15:27
Campylobacteriosis	17/08/2020 15:27
Chickenpox	17/08/2020 15:27
Chikungunya	17/08/2020 15:27
Chlamydiosis	17/08/2020 15:27

Within each of these topics you have a year (e.g. 2020) and then a compressed json file containing the tweets by day for each year. The dates pertain to the dates the tweet was collected (not tweeted). There may be more than one file for one day if the file exceeds 100 MB.



- geolocated
- search
  - Anthrax
    - 2020
  - Antimicrobial resi
  - Avian influenza

Name

- 2020.08.17.00001.json.gz
- 2020.08.18.00001.json.gz
- 2020.08.19.00001.json.gz
- 2020.08.20.00001.json.gz
- 2020.08.21.00001.json.gz

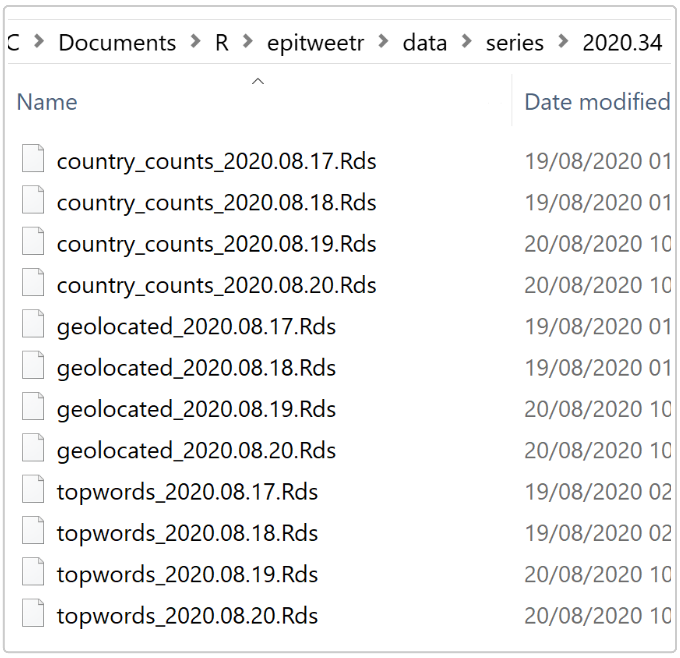
The *geolocated folder* contains compressed json files with geolocated information produced by the geolocated algorithm.

Data folder > series

In the *series* folder, `epitweetr` stores the aggregated data of the geolocated tweets as well as the top words.

There is a folder for each ISO week of date of collection containing Rds files (a native R format file) for each day and series:

- geolocated_YYY.MM.DD.Rds contains daily tweet counts at finest possible location level
- topwords_YYY.MM.DD.Rds contains daily tweet top words counts at country level
- country_counts_YYY.MM.DD.Rds contains hourly tweet counts at country level



C › Documents › R › epitweetr › data › series › 2020.34

Name	Date modified
country_counts_2020.08.17.Rds	19/08/2020 01
country_counts_2020.08.18.Rds	19/08/2020 01
country_counts_2020.08.19.Rds	20/08/2020 10
country_counts_2020.08.20.Rds	20/08/2020 10
geolocated_2020.08.17.Rds	19/08/2020 01
geolocated_2020.08.18.Rds	19/08/2020 01
geolocated_2020.08.19.Rds	20/08/2020 10
geolocated_2020.08.20.Rds	20/08/2020 10
topwords_2020.08.17.Rds	19/08/2020 02
topwords_2020.08.18.Rds	19/08/2020 02
topwords_2020.08.19.Rds	20/08/2020 10
topwords_2020.08.20.Rds	20/08/2020 10

This is the aggregate information as described in the section “How does it work? General architecture of epitweetr > Aggregation”.

The interactive user application (Shiny app)

You can launch the `epitweetr` interactive user application (Shiny app) from the R session by typing in the R console (replace “data_dir” with the desired data directory):

```
epitweetr_app("data_dir")
```

Alternatively, you can use a launcher: In an executable bat or sh file put the following content, (replacing “data_dir” with the expected data directory)

R –vanilla -e epitweetr::epitweetr_app(‘data_dir’)

The `epitweetr` interactive user application has five pages:

- The dashboard, where a user can visualise and explore tweets
- The configuration page, where you can change settings and check the status of the underlying processes
- The alerts page, where you can view the current alerts and associated information
- The geotag evaluation page, where you can evaluate the geolocation algorithm in different tweet fields to manually choose the geolocation threshold
- The troubleshoot page, with with automatic checks and hints for using `epitweetr` with all its functionalities

Dashboard: The interactive user interface for visualisation

The dashboard is where you can interactively explore visualisations of tweets. It includes a line graph (trend line) with alerts, a map and top words of tweets for a given topic. Please note that the first time a period is selected, you will need to **wait 1-2 minutes until you see the outputs**. Likewise, for any new selection (adding a region, changing topic, etc), `epitweetr` will start reading the corresponding data; so if several selections are done, you may need to wait 1-2 minutes until the last selection is shown in the dashboard. When `epitweetr` is reading new data, the outputs have less intensity giving you an indication that new data is being read and plotted.

In order to interactively explore the data, you can select from several filters, such as topics, countries and regions, time period, time unit, signal confidence and days in baseline.

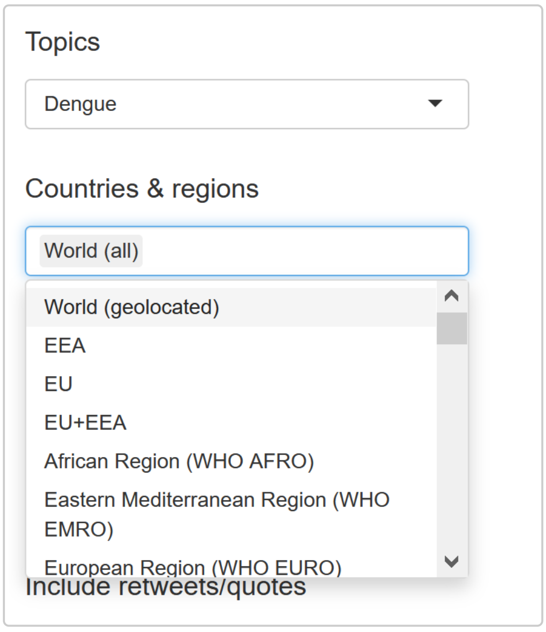
Note that whatever options/settings you select on the dashboard, will have no effect on the alert detection. The alert detection settings are all selected on the configuration page of the Shiny app.

Filters

Topics

You can select one item from the drop-down list of topics, which is populated by what is specified in the topics on the configuration page. You can also start typing in the text field and select the topics from the filtered dropdown list.

Countries & regions


Topics

Dengue

Countries & regions

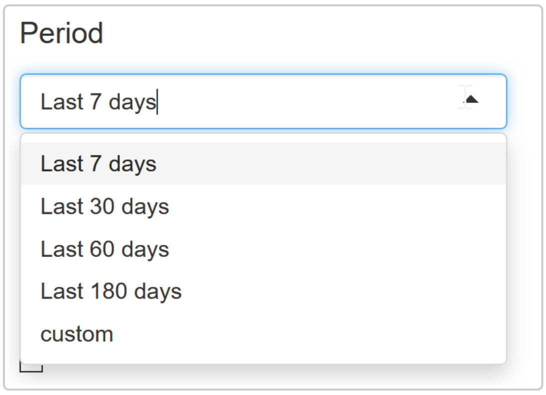
World (all)

- World (geolocated)
- EEA
- EU
- EU+EEA
- African Region (WHO AFRO)
- Eastern Mediterranean Region (WHO EMRO)
- European Region (WHO EURO)

Include retweets/quotes

If you select **World (all)**, all tweets are displayed regardless of their geolocation. You can select an individual country, you can select regions and subregions, and you can select several items at the same time. You can also start typing in the text field and select the geographical item from the drop-down list.

Period

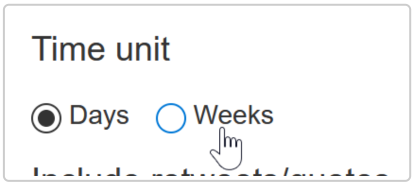

Period

Last 7 days

- Last 7 days
- Last 30 days
- Last 60 days
- Last 180 days
- custom

You can select from the past 7 (the default), 30, 60 or 180 days. You can also select “custom” and a calendar option to select study period will appear. These periods will be the time period for inclusion in the visualisations. When selecting custom period, please ensure that the first date is at least one day before the second date.

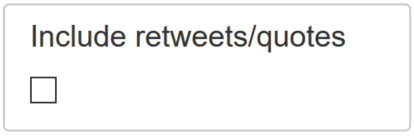
Time unit


Time unit

[✓ Days] Weeks

You can display the timeline for the number of tweets with weeks or days as units of time. The default is days.

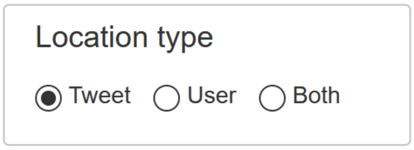
Include Retweets/quotes


Include retweets/quotes

☐

By default, retweets are not included in any of the visualisations. If “include retweets/quotes” is checked, the visualisations display results of tweets and retweets/quotes. Otherwise, the visualisations display only tweets (without retweets/quotes).

Location type

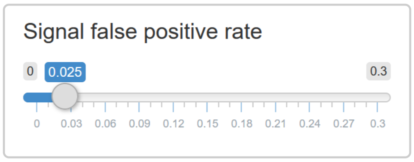

Location type

[✓ Tweet] User Both

Tweets are geolocated into regions, subregions and countries. “Location type” indicates what should be used for geolocation:

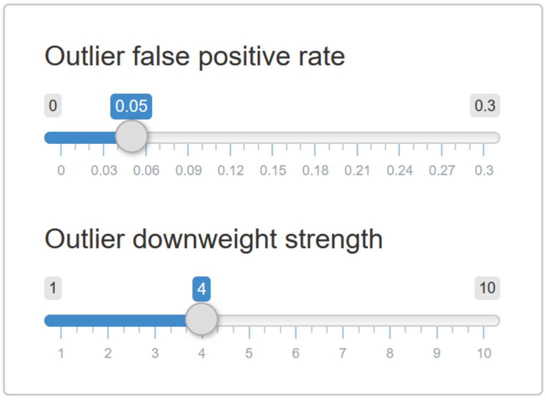
- Tweet: this comprises geographical information contained within the tweet text or if not available, geographical information contained within the retweet/quoted text, if applicable.
- User: this is geographical information obtained from the user location. In order of priority, this is the user’s location at time of the tweet, the user’s API location or the “home” in the public profile if none of these is available.
- Both: the geographical information used for a tweet will be, in order of priority, the location within the tweet text, but if not available the user location.

Signal detection false positive rate


Signal false positive rate: 0.025

Using the slider, you can explore the differences in the signals generated when changing the alpha parameter for the false positive rate. Note that this will not change the signal false positive rate for the alert emails. This is just a tool for the user to explore this parameter. The default is 0.025. A higher false positive rate will increase the sensitivity and possibly the number of signals detected, and vice versa.

Outlier false positive rate and outlier downweight strength

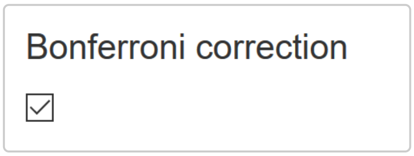

Outlier false positive rate: 0.05

Outlier downweight strength: 4

The outlier false positive rate relates to the false positive rate for determining what an outlier is when downweighting previous outliers/signals. The lower the value, the fewer previous outliers will potentially be included. A higher value will potentially include more previous outliers.

The outlier downweight strength determines how much an outlier will be downweighted by. The higher the value the greater the downweighting. For more information please see Annex I.

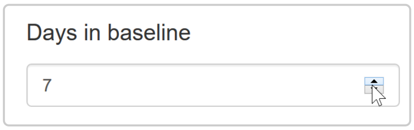
Bonferroni correction


Bonferroni correction ☑

The Bonferroni correction is selected by default. It accounts for false positive signal detection through multiple testing. For country-specific signal detection, the alpha is divided by the number of countries. For continent-specific signal detection, the alpha is divided by the number continents.

If you do not wish to use this correction, you can uncheck it.

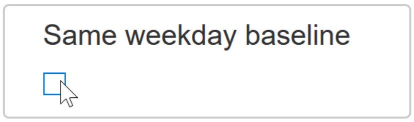
Days in baseline


Days in baseline

7

The default days in baseline is 7. The user can explore the effect of having different days in the baseline. This is only for the visualisation, any changes made for the email alerts have to be made in the configuration page.

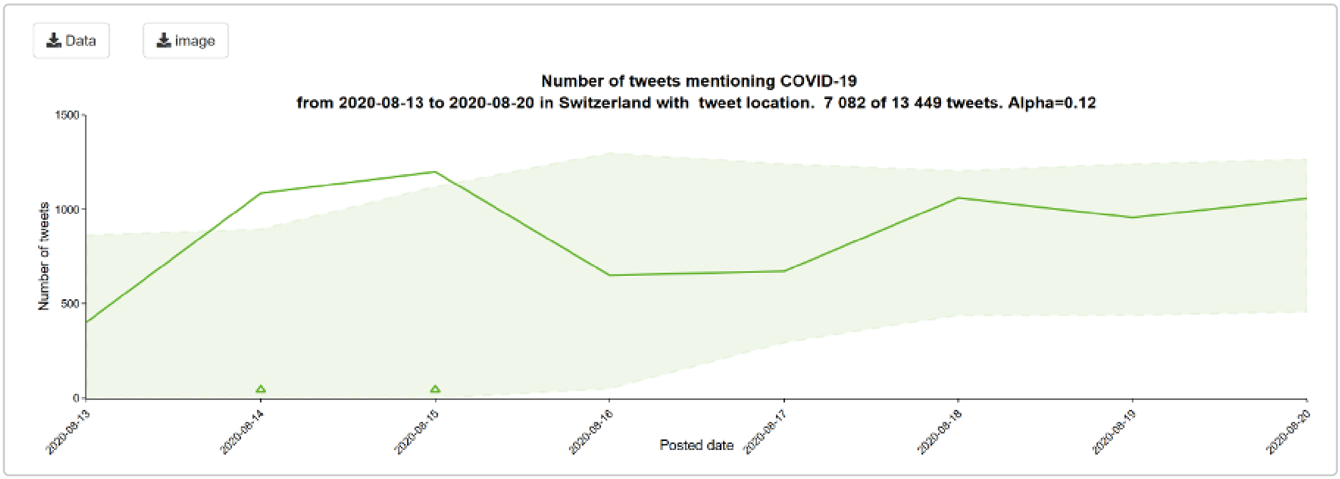
Same weekday baseline


Same weekday baseline

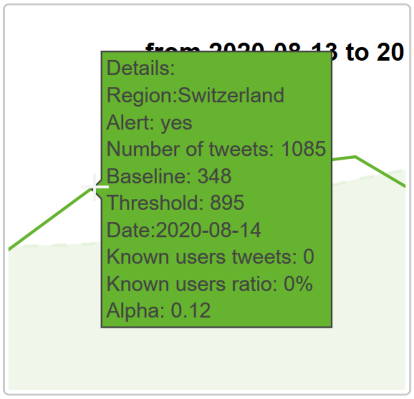
It is possible that there is a "day of the week effect", where more tweets may be tweeted on a given day of the week (e.g. Monday) than on other days. You can also select to calculate the baseline not on consecutive days, but on the past N days that correspond to the same 24 hour window N days back. This way if N = 7, the baseline is calculated using the "days" from -7, -14, -21, -28, -35, -42, -49 and -56 (if the current "day" is zero).

The timeline

The timeline graph is a time series, where you can see the number of tweets for a given topic, geographical unit and study period. Signals are indicated as triangles on the graph, with the alpha and baseline days as specified in the filters. The area under the threshold is indicated in the shaded green colour. Note that the signals are related to the choice of alpha and days in baseline in the filters on the dashboard, rather than what is used for the alert emails. This way you can explore the effect of changing these parameters and adapt the settings for the alert emails if needed.

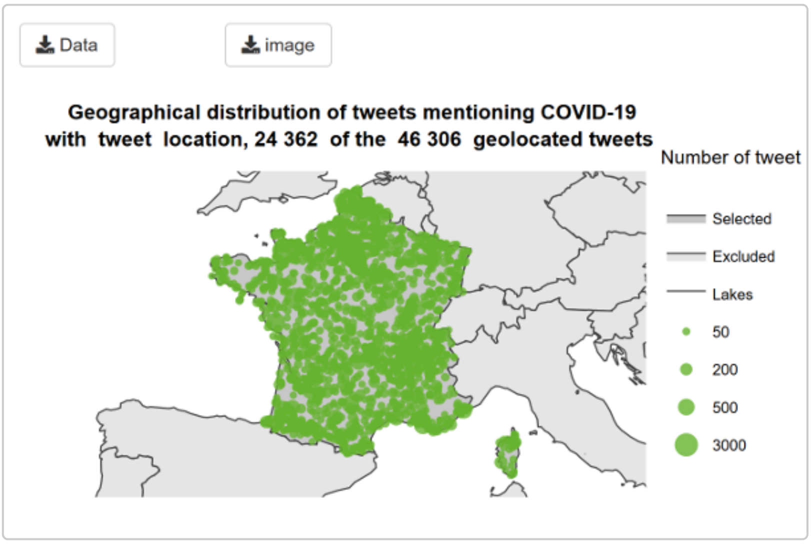


If you hover over the graph, you obtain extra information on country, date, number of tweets and the number of tweets from the list of known users, the ratio of known users to unknown users, whether the number of tweets was associated with a signal and what the threshold and the alpha was.


from 2020-08-13 to 20

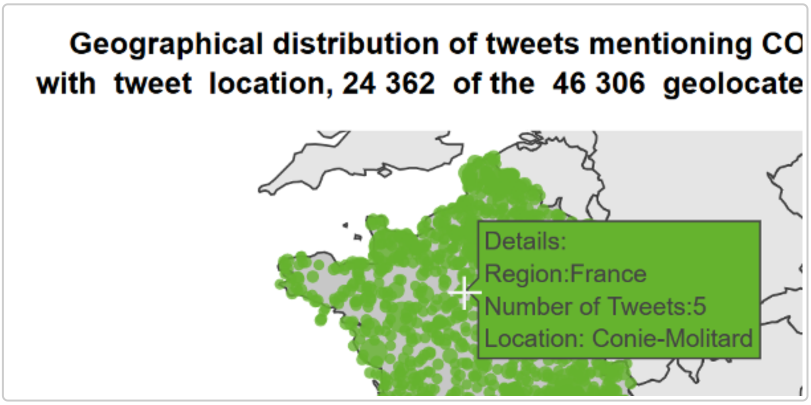
Details:
Region:Switzerland
Alert: yes
Number of tweets: 1085
Baseline: 348
Threshold: 895
Date:2020-08-14
Known users tweets: 0
Known users ratio: 0%
Alpha: 0.12

Map


Data
image
Geographical distribution of tweets mentioning COVID-19
with tweet location, 24 362 of the 46 306 geolocated tweets
Number of tweet
Selected
Excluded
Lakes
50
200
500
3000

The map shows a proportional symbol map of the tweets by country and by topic for the study period. The larger the circle, the greater the number of tweets.

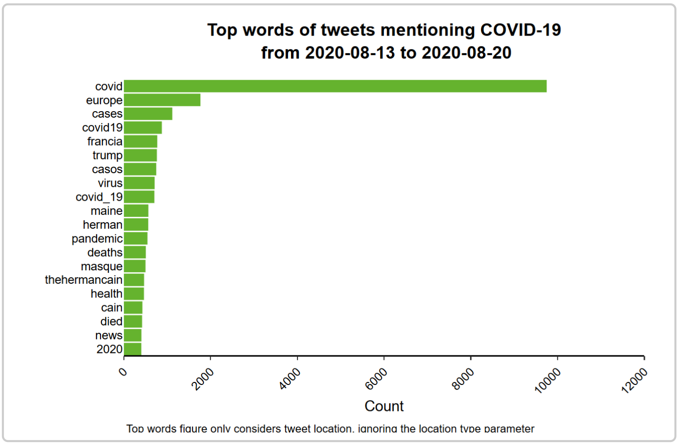
The geographical information for the map is based on the choice in the filters: the country/region/subregion and the location type (tweet, user or both).


Geographical distribution of tweets mentioning CO
with tweet location, 24 362 of the 46 306 geolocate
Details:
Region:France
Number of Tweets:5
Location: Conie-Molitard

When you hover over the map, you can get information on number of tweets and names of the geographical units underlying the circles on the map.

When selecting one country, the symbols show the geographical distribution of tweets at subnational level. When selecting two or more countries or other geographical entity (e.g. regions or continents), the symbols show the geographical distribution of tweets at national level. Note that if a tweet has a geotag at country level (e.g. France), it will not be displayed when selecting only that country since no subnational geotagging is available.

Most frequent words found in tweets



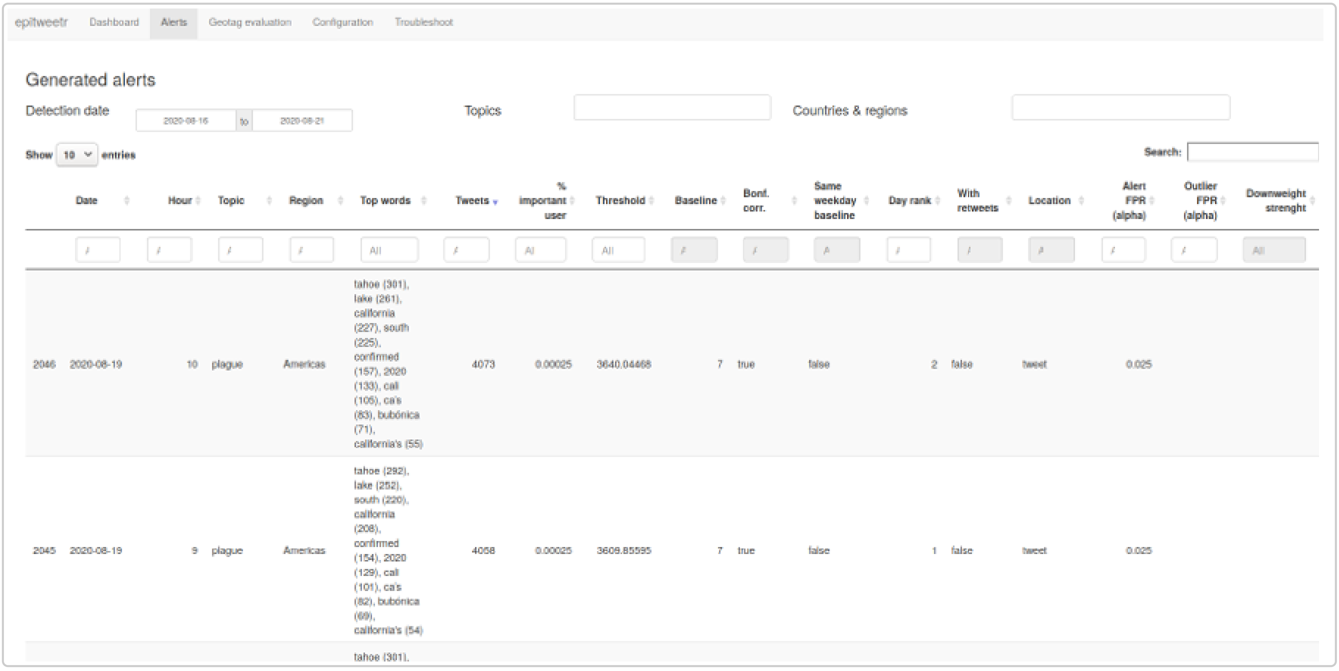
Tweets are analysed globally to get the top 500 words of each chunk for the same language, day and topic. These 500 words are then subsetted by country.

This graph then displays the top words of tweets by topic for the study period for the geographical units chosen, and according to the filter on tweet/retweet.

Note that, differently to other visualisation figures, the most frequent words found in tweets is always related to “tweet location” and is not influenced by the filter option for choice of location (user or tweet location).

The alerts page

The alerts page summarises the signals detected within the specified study period. This summary includes the date, hour, topic and geographical unit of the signal, which top words are included, the number of tweets, the number of tweets from important users and the threshold. It additionally includes many of the settings specified in the configuration page, which were used for detecting signals. This is also the output received in the emails alerts.



Generated alerts

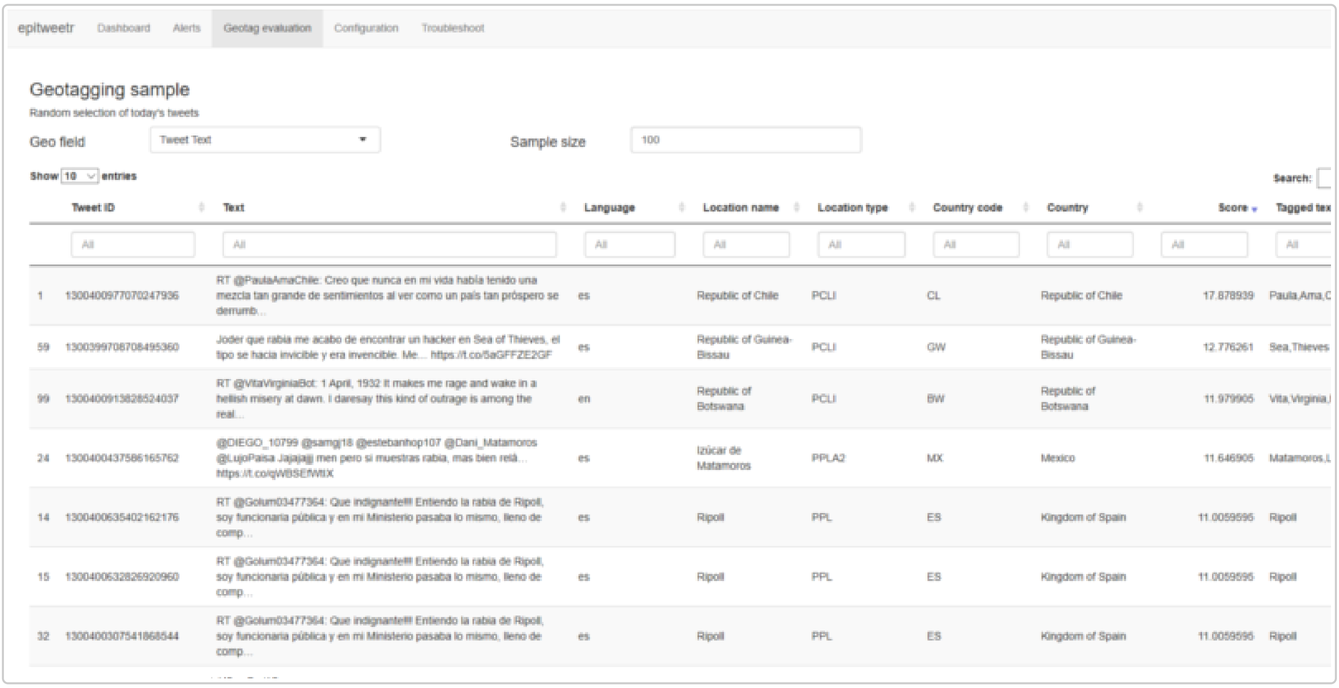
Detection date 2020-08-16 to 2020-08-21 Topics Countries & regions

Show 10 entries Search:

	Date	Hour	Topic	Region	Top words	Tweets	% important user	Threshold	Baseline	Bonf. corr.	Same weekday baseline	Day rank	With retweets	Location	Alert FPR (alpha)	Outlier FPR (alpha)	Downweight strenght
					All		All	All									All
2046	2020-08-19	10	plague	Americas	tahoe (301), lake (261), california (227), south (225), confirmed (157), 2020 (133), cali (105), ca's (83), bubónica (71), california's (55)	4073	0.00025	3640.04468	7	true	false	2	false	tweet	0.025		
2045	2020-08-19	9	plague	Americas	tahoe (292), lake (252), south (220), california (208), confirmed (154), 2020 (129), cali (101), ca's (82), bubónica (69), california's (54)	4058	0.00025	3609.85595	7	true	false	1	false	tweet	0.025		
					tahoe (301),												

The geotag evaluation page

This page supports the user in establishing the geolocation threshold in the configuration page. The user can choose the field of the tweets to test and the number of tweets to sample. This page is for visualization only and no changes in the geolocation detected by `epitweetr` is allowed.



Geotagging sample

Random selection of today's tweets

Geo field: Tweet Text

Sample size: 100

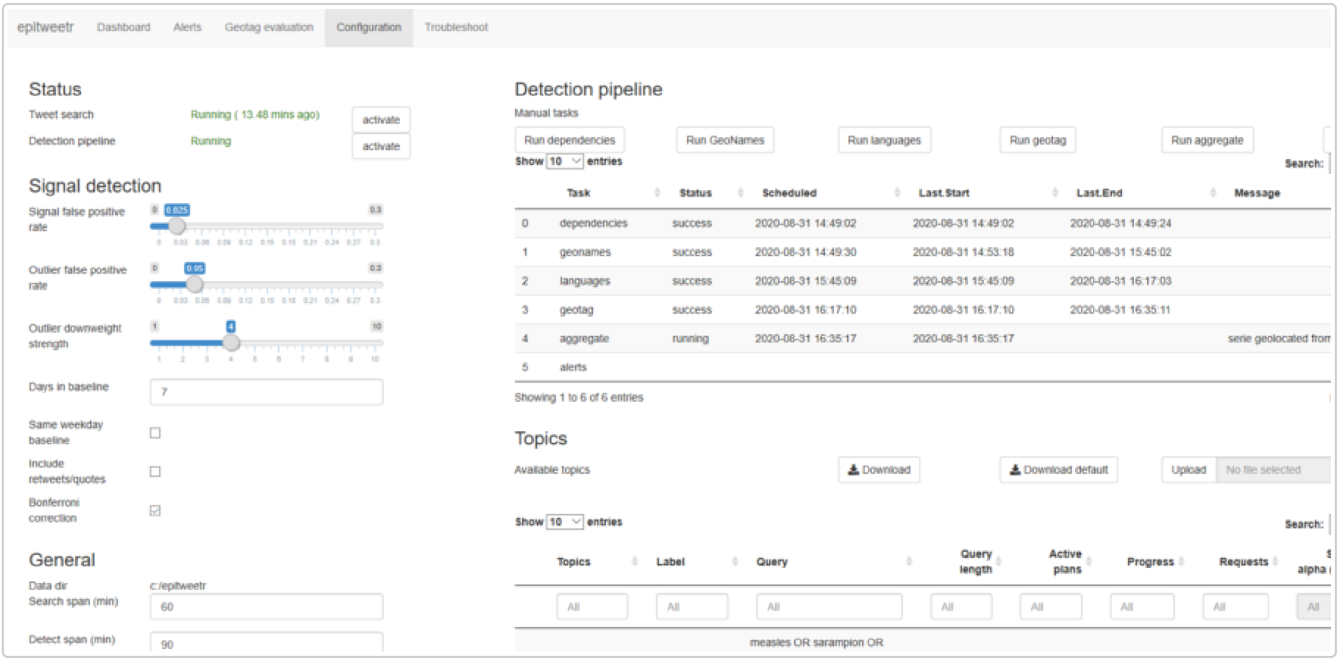
Show 10 entries

Search:

	Tweet ID	Text	Language	Location name	Location type	Country code	Country	Score	Tagged tex
	All	All	All	All	All	All	All	All	All
1	1300400977070247936	RT @PaulaAmaChile: Creo que nunca en mi vida había tenido una mezcla tan grande de sentimientos al ver como un país tan próspero se derrumb…	es	Republic of Chile	PCLI	CL	Republic of Chile	17.878939	Paula,Ama,C
59	1300399708708495360	Joder que rabia me acabo de encontrar un hacker en Sea of Thieves, el tipo se hacia invicible y era invencible. Me… https://t.co/5aGFFZE2GF	es	Republic of Guinea-Bissau	PCLI	GW	Republic of Guinea-Bissau	12.776261	Sea,Thieves
99	1300400913828524037	RT @VitaVirginiaBot: 1 April, 1932 It makes me rage and wake in a hellish misery at dawn. I daresay this kind of outrage is among the real…	en	Republic of Botswana	PCLI	BW	Republic of Botswana	11.979905	Vita,Virginia,I
24	1300400437586165762	@DIEGO_10799 @samgj18 @estebanhop107 @Dani_Matamoros @LujoPaisa Jajajajj men pero si muestras rabia, mas bien reià… https://t.co/qWBSEfWtIX	es	Izúcar de Matamoros	PPLA2	MX	Mexico	11.646905	Matamoros,L
14	1300400635402162176	RT @Golum03477364: Que indignante!!! Entiendo la rabia de Ripoll, soy funcionaria pública y en mi Ministerio pasaba lo mismo, lleno de comp…	es	Ripoll	PPL	ES	Kingdom of Spain	11.0059595	Ripoll
15	1300400632826920960	RT @Golum03477364: Que indignante!!! Entiendo la rabia de Ripoll, soy funcionaria pública y en mi Ministerio pasaba lo mismo, lleno de comp…	es	Ripoll	PPL	ES	Kingdom of Spain	11.0059595	Ripoll
32	1300400307541868544	RT @Golum03477364: Que indignante!!! Entiendo la rabia de Ripoll, soy funcionaria pública y en mi Ministerio pasaba lo mismo, lleno de comp…	es	Ripoll	PPL	ES	Kingdom of Spain	11.0059595	Ripoll

The configuration page

In the configuration page, you can change settings of the tool, you can check the status of the various processes/pipelines of the tool and you can add, delete and modify topics and their associated requests, languages for geolocation and the list of the “important users” and email alert subscribers. When changing anything in the “Signal detection” or “General” sections, do not forget to click on the “Update Properties” button at the end of the “General” section. The following sections describe the configuration page in more detail.



Status

Tweet search	Running (13.48 mins ago)	activate
Detection pipeline	Running	activate

Signal detection

Signal false positive rate	0.025 (0 – 0.3)
Outlier false positive rate	0.05 (0 – 0.3)
Outlier downweight strength	4 (1 – 10)
Days in baseline	7
Same weekday baseline	☐
Include retweets/quotes	☐
Bonferroni correction	☑

General

Data dir	c:/epitweetr
Search span (min)	60
Detect span (min)	90

Detection pipeline

Manual tasks

Show 10 entries

Search:

	Task	Status	Scheduled	Last.Start	Last.End	Message
0	dependencies	success	2020-08-31 14:49:02	2020-08-31 14:49:02	2020-08-31 14:49:24	
1	geonames	success	2020-08-31 14:49:30	2020-08-31 14:53:18	2020-08-31 15:45:02	
2	languages	success	2020-08-31 15:45:09	2020-08-31 15:45:09	2020-08-31 16:17:03	
3	geotag	success	2020-08-31 16:17:10	2020-08-31 16:17:10	2020-08-31 16:35:11	
4	aggregate	running	2020-08-31 16:35:17	2020-08-31 16:35:17		serie geolocated from
5	alerts					

Showing 1 to 6 of 6 entries

Topics

Available topics

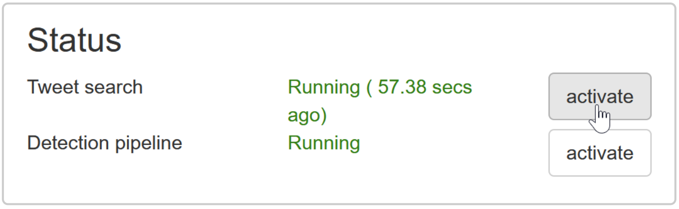
Show 10 entries

Search:

Topics	Label	Query	Query length	Active plans	Progress	Requests	alpha
All	All	All	All	All	All	All	All
		measles OR sarampion OR					

Status

The status section enables you to quickly assess the latest time point and/or status of the processes for tweet collection (Tweet Search), and geolocation, aggregation and signal detection (Detection pipeline).



Status

Tweet search	Running (57.38 secs ago)	activate
Detection pipeline	Running	activate

In the status section, you can tell if both the search pipeline and the detect pipeline processes are running. If you are running Windows, you can click on “activate” to register these processes as scheduled tasks and run them manually from the Windows task scheduler.

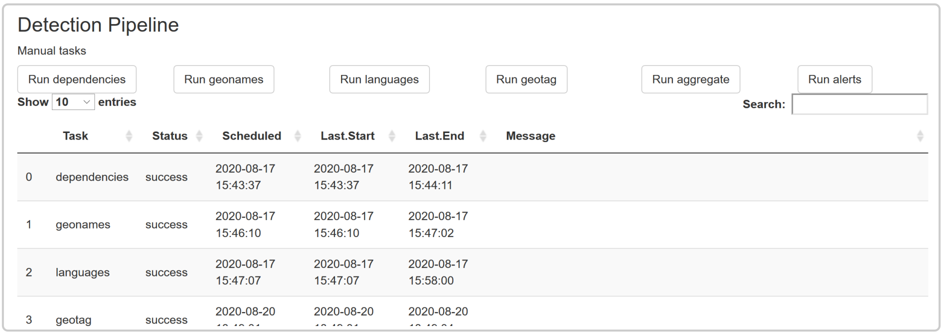
Detection pipeline

In the manual tasks in the “Detection pipeline”, you have. You need to run manually the tasks of dependencies, geonames and languages by clicking in the buttons “Run dependencies”, “Run geonames” and “Run languages” the first time you use `epitweetr`, and then only if you are downloading new versions. Geonames and languages relate to the geolocation and language models used by epitweetr. If you would like to update them (this is not something that needs to be done regularly, more on a yearly basis or so), then you can click on “Run”.

The “Run geotag”, “Run aggregate” and “Run alerts” buttons can be used to force the start of these tasks in case there is any error or issue. You can check their status in the “Detection Pipeline” table.

The detection pipeline gives more information about the status of the processes of `epitweetr`. This is useful for troubleshooting any issues arising and monitoring progress. It contains the five tasks that are running in the background. Geoames and languages are tasks that will download and update the local copies of these. This will only be triggered if we add a language or update GeoNames. The start and end dates will generally be much older than those of geotag, aggregate and alerts.

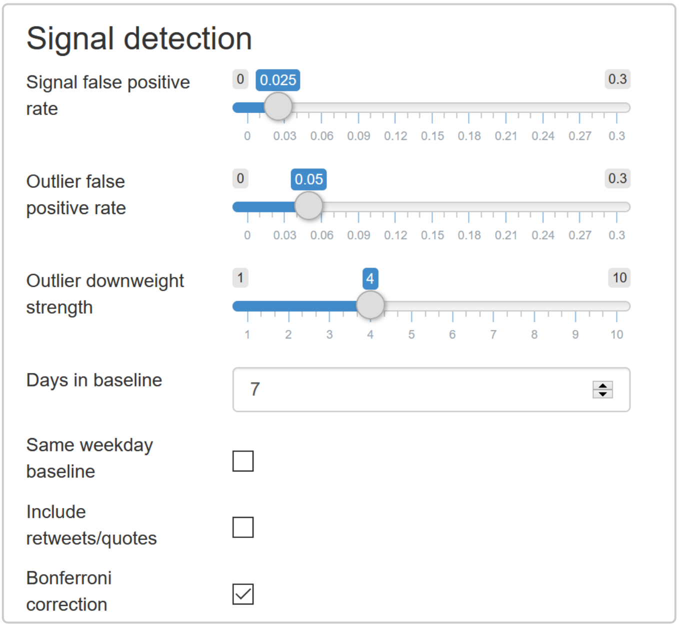
Geotag, aggregate and alerts dates should be more recent if the search and detection pipelines are active and running. They are scheduled according to the detect span. The status can include running, scheduled, pending, failed or aborted (if it has failed more than three times).



	Task	Status	Scheduled	Last.Start	Last.End	Message
0	dependencies	success	2020-08-17 15:43:37	2020-08-17 15:43:37	2020-08-17 15:44:11	
1	geonames	success	2020-08-17 15:46:10	2020-08-17 15:46:10	2020-08-17 15:47:02	
2	languages	success	2020-08-17 15:47:07	2020-08-17 15:47:07	2020-08-17 15:58:00	
3	geotag	success	2020-08-20	2020-08-20	2020-08-20	

Signal detection

In the signal detection section in the configuration page, you can set the signal false positive rate alpha parameter, which increases (if larger) the the detection interval (more signals are detected), or decreases (if smaller) the detection interval (fewer signals are detected).



Signal false positive rate: 0.025
Outlier false positive rate: 0.05
Outlier downweight strength: 4
Days in baseline: 7
Same weekday baseline: ☐
Include retweets/quotes: ☐
Bonferroni correction: ✓

The outlier false positive rate relates to the false positive rate for determining what an outlier is when downweighting previous outliers/signals. The lower the value, the fewer previous outliers will potentially be included. A higher value will potentially include more previous outliers.

The outlier downweight strength determines how much an outlier will be downweighted by. The higher the value the greater the downweighting. For more information please see Annex I.

`epitweetr` calculates a threshold to determine if the current number of tweets for a given 24-hour window exceeds what is expected (see section *"How does it work? General architecture behind epitweetr > Signal detection"*). This threshold is based on a default of the previous 7 days. In the "default days in baseline" field, you can change the number of days.

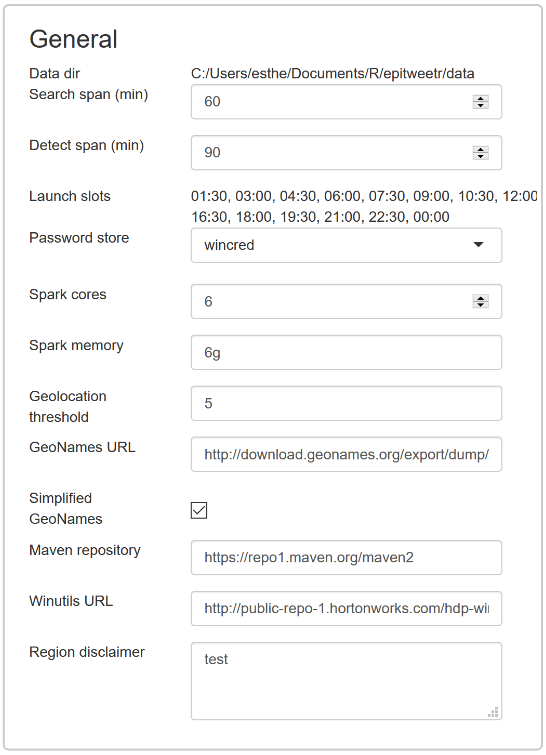
You can also change the default of using the previous 7 days for calculating a baseline to the previous 7 same days of the week, in order to avoid a "day of the week effect" (it may be that there are always more tweets about this topic on a Monday, for example, which could affect the signal detection).

You can also specify if the signal detection is carried out just with tweet text, or includes retweets/quotes (check the box "Default with retweets/quotes").

The last checkbox "Default with Bonferroni correction", take multiple testing into account, which can result in false positives. If this box is checked then the signal detection alpha parameter is divided by the number of geographical locations in which signal detection is carried out. For example, at country level, the alpha parameter is divided by the total number of countries. At continent level, the alpha parameter is divided by the total number of continents.

When changing anything in the "Signal detection" section, do not forget to click on the "Update Properties" button at the end of the "General" section.

General



General

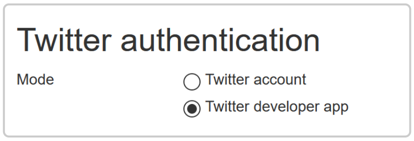
Data dir	C:/Users/esthe/Documents/R/epitweetr/data
Search span (min)	60
Detect span (min)	90
Launch slots	01:30, 03:00, 04:30, 06:00, 07:30, 09:00, 10:30, 12:00 16:30, 18:00, 19:30, 21:00, 22:30, 00:00
Password store	wincred
Spark cores	6
Spark memory	6g
Geolocation threshold	5
GeoNames URL	http://download.geonames.org/export/dump/
Simplified GeoNames	✓
Maven repository	https://repo1.maven.org/maven2
Winutils URL	http://public-repo-1.hortonworks.com/hdp-wi
Region disclaimer	test

- In **Data directory** you can view the directory that `epitweetr` uses to store the tweet and associated data collected. This is also the directory that the dashboard uses to obtain the datasets for displaying the visualisations. You need to set this folder when you launch `epitweetr` or set the environment variable 'EPI_HOME'.
- The **Search span** relates to how long a search plan is carried out. The default is 60 minutes. This value controls the size of the search window of tweets. If you reduce this value you will get tweets sooner, but you may 'waste' requests on topics with very few tweets. If you increase its value you will take more time to get the tweets but you will get more requests for popular tweets increasing the chances of being exhaustive. You can see when you are not able to collect the tweets on the Shiny configuration page if you have more than one active plan for some topics.
- The **Detect span** relates to how frequently the processes of the detection pipeline (geotagging, aggregation and alert detection) are carried out. The default is 90 minutes. Email alerts are sent at the end of the detect loop. This value is treated as a lower bound, the detect loop could take more time to finish depending on the volume of tweeets and your system specifications.

- The **Launch slots** for the detection pipeline processes will be spaced out according to the “Detect span”, with the first one starting at midnight. These values can be used in the subscribers file of the configuration page.
- To avoid storing Twitter credentials in plain files, `epitweetr` uses a system dependent password store functionality, which is stored in the **Password store**. Depending on your system you can choose the mechanism that suits the environment where `epitweetr` is running. For details on each implementation see [*https://CRAN.R-project.org/package=keyring*](https://CRAN.R-project.org/package=keyring)
    - wincred: (Windows only) uses the windows credential manager.
    - macos: (MAC only) uses the Mac OS keychain services
    - file: Uses password protected encrypted files
    - secret service: (Linux only) uses Linux secret service
    - environment: Uses environment variables (extra setup needed, see [*https://CRAN.R-project.org/package=keyring*](https://CRAN.R-project.org/package=keyring))
- **Spark cores and spark memory:** The memory allocation for `epitweetr` in terms of CPU (Spark cores) and RAM (Spark Memory) is also defined in the “general” section. The default is 6 cores and 6 GB of RAM. This will depend on the CPU and RAM capacity of your machine and it has to be equal or less than that.
- **Geolocation threshold:** During the geolocation process, sets of words are processed and potential matches to existing locations are determined and given a score. The higher the score, the greater the probability that the geolocation is correct. A threshold is set in `epitweetr`, under which any matches are not considered good enough for geolocation. The scale goes from 1 to 10, and the default is set at 5.
- **Geonames URL:** The URL used to download the GeoNames database (used for generating locations) is in the general section. Should this URL ever change, you can make the amendment here.
- **Simplified geonames:** As GeoNames is a very large file, a simplified version of it is used by default, including only existing geographical locations where the population is known. You can uncheck this option if you wish to use the whole GeoNames database.
- **Maven repository:** This is the URL of the maven repository that will be used to download the JAR dependencies for the detect loop, mainly Spark and Lucene.
- **Winutils URL:** This is the URL that will be used to download winutils.exe. This is a Windows binary necessary for running Spark locally on Windows. If you do not want to use this version you can produce it yourself by downloading Hadoop 2.8.4 or higher and compiling it on a Windows machine.
- **Region disclaimer:** If you would like to add a disclaimer to the map you are using. This disclaimer is added to the image export of the dashboard map, and also the PDF export of the dashboard.

Twitter authentication

You have two options for authentication to collect tweets, using a Twitter account (utilising the rtweet package) and using a Twitter developer application. You can select which option you will use in the **Twitter authentication** section. See section *“How does it work? General architecture behind epitweetr > Collection of tweets > Twitter authentication”* for more details on how to do the Twitter authentication.


Twitter authentication

Mode: ○ Twitter account [✓ ● Twitter developer app]

Email authentication (SMTP)

In this section, you need to specify the email authentication (SMTP) details for the email that will send the alerts.

If **Unsafe certificates** is checked, then `epitweetr` will use your SMTP server even if the server sends an invalid certificate.

When changing anything in the “general” section, do not forget to click on the “Update Properties” button.

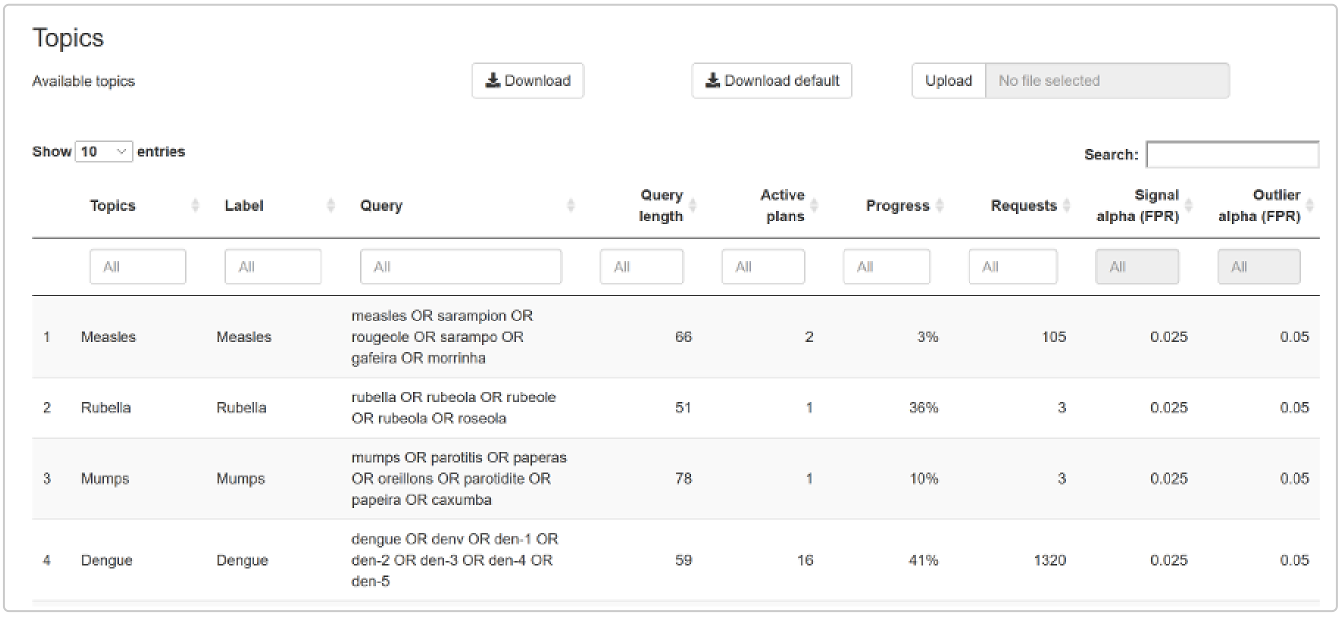
Topics

Topics are what determines what tweets `epitweetr` collects. This is done via an Excel spreadsheet that contains the topics and the associated requests that `epitweetr` uses to query Twitter API with.

A query consists of keywords and operators that are used to match on tweet attributes. See section *"How does it work? General architecture behind epitweetr > Collection of tweets > Topics of tweets to collect and queries"* for more details about queries.

`epitweetr` comes with a default list of topics as used by the ECDC Epidemic Intelligence team at the date of package generation (*1st of September, 2020*). You can download this list of topics and upload your own in the "Available Topics" section in the configuration page. See section *"How does it work? General architecture behind epitweetr > Collection of tweets > Topics of tweets to collect and queries"* for more details on how to structure the topics list.

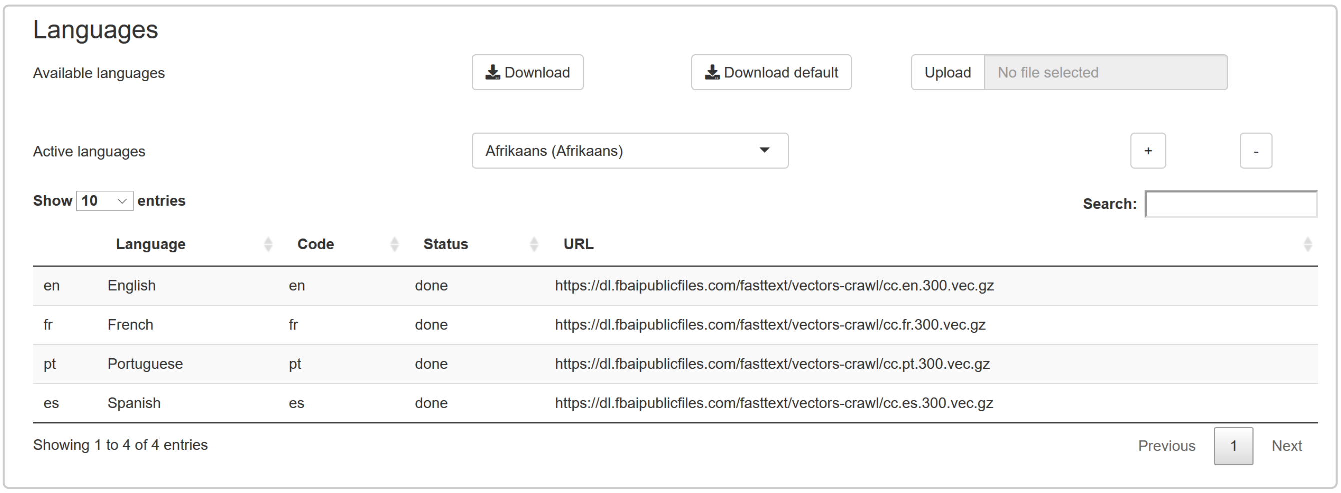
In the topics section on the configuration page, you can view the topic, the associated query, the query length and how many active search plans are associated with the query. If more than one search plan is active, this means that `epitweetr` did not manage to collect all possible tweets in the last session. Additionally, you can see the progress and the number of requests from the last search plan.



	Topics	Label	Query	Query length	Active plans	Progress	Requests	Signal alpha (FPR)	Outlier alpha (FPR)
	All	All	All	All	All	All	All	All	All
1	Measles	Measles	measles OR sarampion OR rougeole OR sarampo OR gafeira OR morrinha	66	2	3%	105	0.025	0.05
2	Rubella	Rubella	rubella OR rubeola OR rubeole OR rubeola OR roseola	51	1	36%	3	0.025	0.05
3	Mumps	Mumps	mumps OR parotitis OR paperas OR oreillons OR parotidite OR papeira OR caxumba	78	1	10%	3	0.025	0.05
4	Dengue	Dengue	dengue OR denv OR den-1 OR den-2 OR den-3 OR den-4 OR den-5	59	16	41%	1320	0.025	0.05

Languages

In the languages section, you can determine which language models are used to identify text during the geolocation process. The default languages are French, English, Portuguese and Spanish. You can download and upload the language models in the "Available Languages" section and add and delete languages used by `epitweetr` in the "Active Languages" section. Please consider the computational cost of adding too many languages, depending on the capacity of your machine.



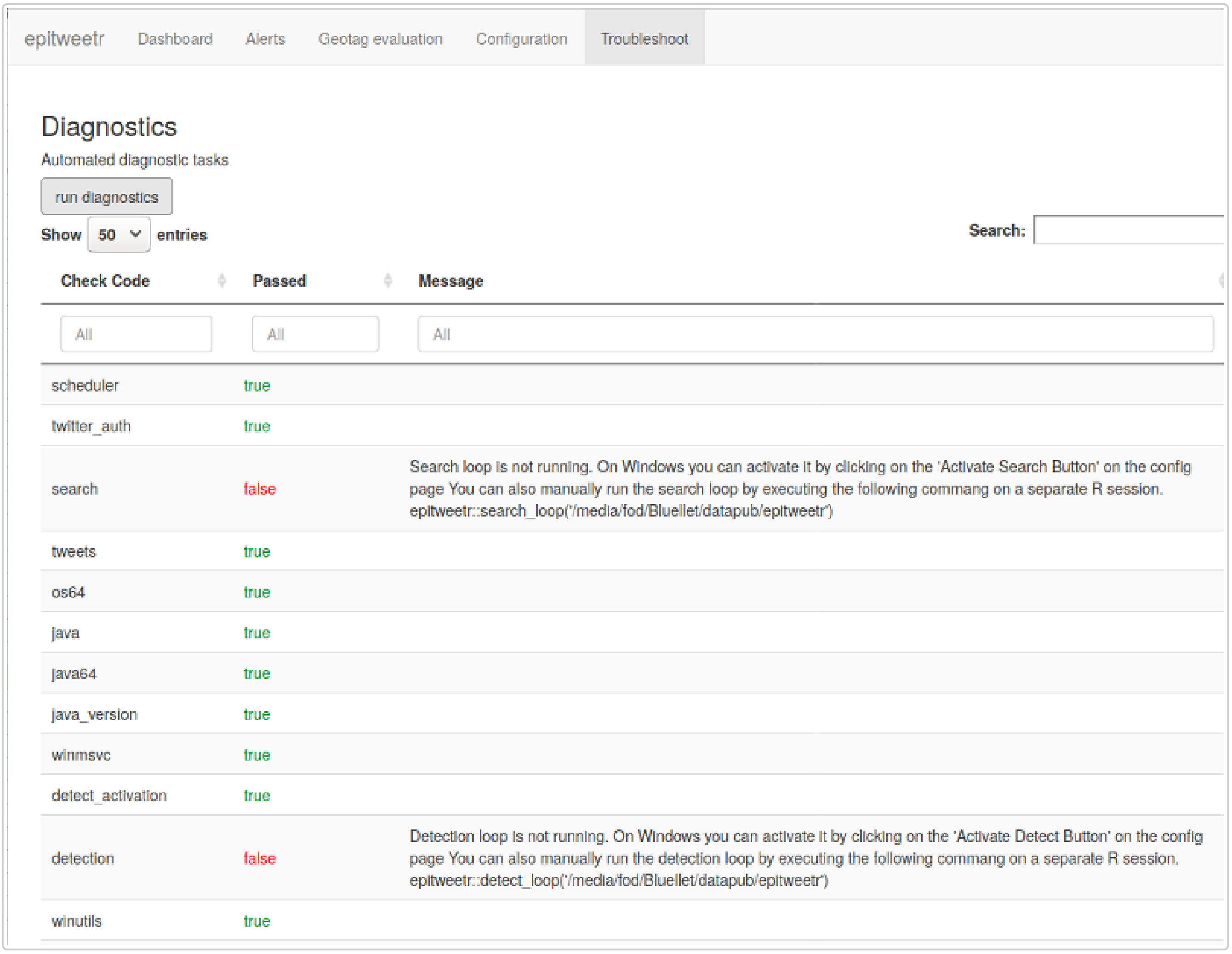
	Language	Code	Status	URL
en	English	en	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.en.300.vec.gz
fr	French	fr	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.fr.300.vec.gz
pt	Portuguese	pt	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.pt.300.vec.gz
es	Spanish	es	done	https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.es.300.vec.gz

Showing 1 to 4 of 4 entries

Previous 1 Next

The troubleshoot page

The troubleshoot page has a list of automatic check and hint for using `epitweetr` with all its functionalities. Click on “Run diagnostics” to see the list of checks, whether it passed the check (“true”) or not (“false”), and hints in case it did not pass the check. More detailed information can be found in Annex II of this user documentation.



Diagnostics

Automated diagnostic tasks

Check Code	Passed	Message
All	All	All
scheduler	true	
twitter_auth	true	
search	false	Search loop is not running. On Windows you can activate it by clicking on the 'Activate Search Button' on the config page You can also manually run the search loop by executing the following commang on a separate R session. epitweetr::search_loop('/media/fod/Bluellet/datapub/epitweetr')
tweets	true	
os64	true	
java	true	
java64	true	
java_version	true	
winmsvc	true	
detect_activation	true	
detection	false	Detection loop is not running. On Windows you can activate it by clicking on the 'Activate Detect Button' on the config page You can also manually run the detection loop by executing the following commang on a separate R session. epitweetr::detect_loop('/media/fod/Bluellet/datapub/epitweetr')
winutils	true	

Downloading outputs from the interactive user interface (Shiny app)

Each visualisation on the Shiny app dashboard can be downloaded as an image, using the “image button”. A png is a portable network graphic file and is a versatile file format for images that do not need to be of a very high resolution (e.g. professional print graphics).

Note that the png format is not supported in the Internet Explorer browser (but you can download a svg file instead).

You can also download the data of each visualisation by clicking on the data button. This will give you a csv file containing the underlying data that you can use for further analysis or to create your own graphs.

Alternatively, you can use the PDF or the Md button at the bottom of the filters to download a PDF or an HTML file of the dashboard Note that for this you will need to have MiKTeX or TinyTeX installed.

Annex I: Downweighting the previous signals

Introduction

In this annex we propose a downweighting approach built as part of the `eears` algorithm used in the `epitweetr` package and which was described above.

Let the $\mathbf{y}$ denote the vector of historic values which is of length n. Part of the computation of the prediction interval at time 0 is the computation of the mean and standard deviation of these historic values, i.e.

$$\overline{y}_0 = \frac{1}{n} \sum_{t=-n}^{-1} y_t \quad \text{and} \quad s_0^2 = \frac{1}{n-1} \sum_{t=-n}^{-1} (y_t - \overline{y}_0)^2$$

The upper limit of the one-sided $(1-\alpha) \times 100\%$ prediction interval for the observation y_0 under an $y_t \overset{\text{iid}}{\sim} N(\mu, \sigma^2), t = -n, \ldots, 0$ model is then computed as

$$U_0 = \overline{y}_0 + t_{1-\alpha}(n-1) \times s_0 \times \sqrt{1 + \frac{1}{n}},$$

where $t_{1-\alpha}(n-1)$ denotes the $1-\alpha$ quantile of the t-distribution with $n-1$ degrees of freedom. This computation of the threshold corresponds to a statistical sound computation of the threshold (Allévius and Höhle 2017).

A desired extension of the above algorithm is the handling of previous signals in the historic values. This problem was already addressed in the quasi-Poisson framework of C. P. Farrington et al. (1996) by first performing a GLM fit and then re-fit the GLM with weights based on the Anscombe residuals. We follow the same general idea, but adapt it to the Gaussian response used in the EARS algorithm and corresponding residuals from the linear model.

EARS as a Linear Model

We first observe that the above estimation of μ and σ^2 through $\overline{y}_0$ and s_0^2 at time 0 can be embedded within a linear regression model, i.e. for $i = 1, \ldots, n$ we model

$$y_i = \mu + \epsilon_i, \quad \text{where} \quad \epsilon_t \overset{\text{iid}}{\sim} N(0, \sigma^2).$$

Note that we, for compatibility with the standard exposition in linear model theory, have indexed the y values s.t. y_{-n} corresponds to y_1 and y_{-1} corresponds to y_n. In matrix terms let $\mathbf{y} = (y_1, \ldots, y_n)'$ and for the intercept-only model the design matrix is $\mathbf{X} = (1, \ldots, 1)'$, which has rank $k = 1$. Thus from standard OLS theory:

$$\hat{\mu} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} = \frac{1}{n} \sum_{i=1}^{n} y_i,$$

which corresponds to $\overline{y}_0$. Furthermore, let the raw residuals be defined as $e_i = y_i - \hat{\mu}$ for $i = 1, \ldots, n$ and denote by $\mathbf{e} = (e_1, \ldots, e_n)'$ the corresponding vector of residuals. Then

$$\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}} = \mathbf{y} - \mathbf{P}\mathbf{y} = (\mathbf{I} - \mathbf{P})\mathbf{y}$$

where $\mathbf{P} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ is the so called hat-matrix known from linear modelling. With this notation we can write up the estimate for σ^2 as in Chatterjee and Hadi (1988):

$$\hat{\sigma}^2 = \frac{\mathbf{e}'\mathbf{e}}{n-k} = \frac{\mathbf{y}'(\mathbf{I} - \mathbf{P})\mathbf{y}}{n-k} = \frac{1}{n-1} \sum_{t=-7}^{1} (y_t - \hat{\mu})^2,$$

which corresponds to the above used expression for s_0^2.

Downweighting

We now compute the so called **externally Studentized residuals** (Chatterjee and Hadi 1988)

$$r_i^* = \frac{e_i}{\hat{\sigma}_{(i)}\sqrt{1-p_{ii}}}, \quad i=1,\ldots,n,$$

where p_{ii} is i'th diagonal element of the hat-matrix $\mathbf{P}$ from the corresponding linear model used above. Furthermore,

$$\hat{\sigma}^2_{(i)} = \frac{\mathbf{y}'_{(i)}(\mathbf{I}-\mathbf{P}_{(i)})\mathbf{y}_{(i)}}{n-k-1}$$

is the variance estimate obtained from a linear regression, where the i'th observation is removed. Linear modelling theory (Chatterjee and Hadi 1988) now states that

$$r_i^* \overset{\text{identical}}{\sim} t(n-k-1).$$

Note that the residuals are only identically distributed, because they are not independent (see Section 4.2.1. of Chatterjee and Hadi (1988) for details). However, the above distributional form allows us to assess for each historic value, if it can be considered as an outlier. For this purpose define $r_{\text{threshold}}$ as the $1-\alpha_{\text{outlier}}$ quantile of the t-distribution with $n-k-1$ degrees of freedom. A historic value is an outlier (for which one possible explanation is that it originates from a true increase in tweets, e.g. an outbreak situation), if $r_i^* > r_{\text{threshold}}$. We shall use this to formulate a weighting schemes for the historic values:

Downweight-Outliers:

$$\begin{aligned} w_i^{(\text{dw})} &= \begin{cases} 1 & \text{if } r_i^* < r_{\text{threshold}} \\ \left(\frac{r_{\text{threshold}}}{(r_i^*)}\right)^k & \text{otherwise} \end{cases} \\ &= \min\left\{1, \left(\frac{r_{\text{threshold}}}{r_i^*}\right)^k\right\}, \end{aligned}$$

where the decay parameter $k>0$ is a known quantity. In the original C. P. Farrington et al. (1996) algorithm, $k=2$ was used. Furthermore, a threshold value of 1 was used. In the later Noufaily et al. (2013) paper, however, a threshold value of 2.58 was recommended. Note: both values are for the standardized Anscombe residuals, which follow a standard normal distribution. If we take corresponding quantiles for the t-distribution with 6 degrees of freedom the values would be 1.09 and 3.72. Note also that the term $(r_{\text{threshold}}/r_i^*)^k$ is a slight adaptation of C. P. Farrington et al. (1996), which instead uses $1/(r_i^*)^2$. The advantage of our proposal is that it ensures a smooth handling of values around the threshold if the threshold is not 1. It might be worth considering a higher power than 2 to ensure an even larger down-weighting for gross outliers. The current default value for the decay parameter in `epitweetr` is 4.

Finally, as in C. P. Farrington et al. (1996), we normalise the weights such that they yield a sum of n by

$$w_i^* = n \times \frac{w_i}{\sum_{i=1}^n w_i}$$

and then re-fit the linear model with these weights. For this purpose define the weight matrix as $\mathbf{W} = \text{diag}(w_1^*,\ldots,w_n^*)$. We can use a subsequent weighted least squares approach to find

$$\hat{\mu}_W = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}\mathbf{y} = \frac{1}{n}\sum_{i=1}^n w_i^* y_i,$$

where the 2nd equal sign is because $(\mathbf{X}'\mathbf{W}\mathbf{X}) = \sum_{i=1}^n w_i = n$ and $\mathbf{X}'\mathbf{W}\mathbf{y} = \sum_{i=1}^n w_i^* y_i$. Furthermore,

$$s_W^2 = \frac{\mathbf{y}'(\mathbf{I}-\mathbf{P}_W)\mathbf{y}}{n-k} = \frac{\sum_{i=1}^n w_i^*(y_i-\mu_W)^2}{n-1},$$

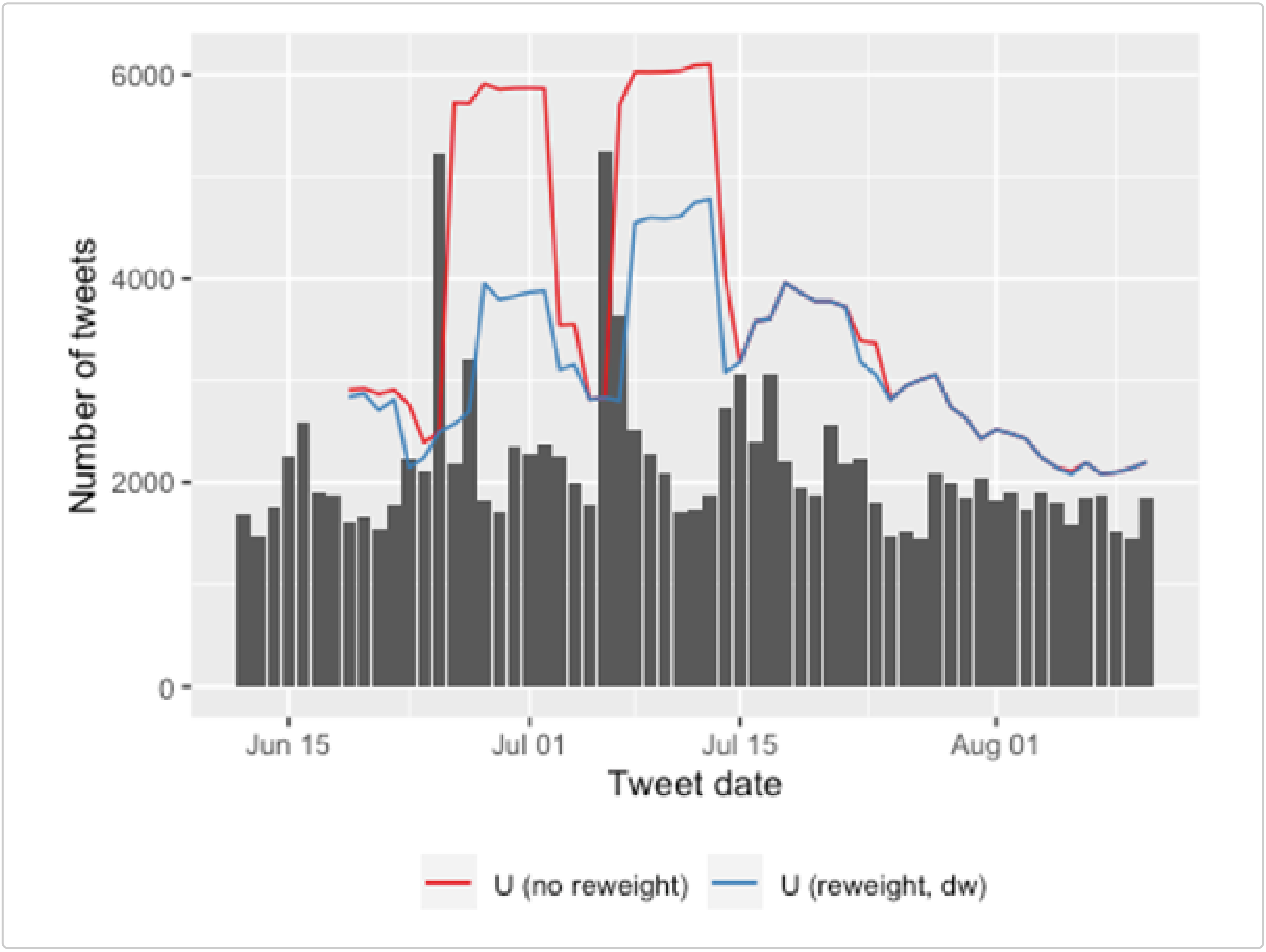
where $P_W = \mathbf{X}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}\mathbf{W}$ is the hat-matrix of the weighted least squares.

The downweighted procedure thus operates with μ_W and s_W^2 instead of $\overline{y}_0$ and s_0^2, respectively, when computing the upper limit U_0 using the above mentioned formula.

Example of the downweighting approach using Ebola data

Figure 5 below shows the upper limit of the signal detection threshold for `epitweetr` Ebola data, both the original (in red) with no downweighting and the downweighted upper threshold (in blue) after taking previous signals in the historic values into account. Note that the downweighted upper threshold detects three additional signals, compared to the original threshold.

Fig 5: Upper limit with and without downweighting for the epitweetr Ebola data



Annex II: Troubleshooting and tips

This annex contains some tips and common solutions to errors or issues that `epitweetr` users may encounter, including an explanation on the checks included in the troubleshoot page.

The troubleshoot page

After running the diagnostics in the troubleshoot page, you can see the checks and status on the following aspects:

- **scheduler**: R package `taskscheduleR` is installed. Only applicable for Windows machines
- **twitter_auth**: Twitter token has been created after authenticating either with Twitter account or Twitter developer application
- **search_running**: search task is running

- **tweets**: tweets have been collected
- **os64**: R is 64bits
- **java**: Java is installed and accessible to epitweetr
- **java64**: Java 64bits is installed and accessible to epitweetr
- **java_version**: the Java version installed is compatible with `epitweetr`
- **winmsvc**: Microsoft Visual C++ 2010 SP1 Redistributable Package is installed. Only applicable for Windows machines
- **detect_activation**: detect loop has been activated
- **detection_running**: detection task is running
- **winutils**: winutils is installed. If false, it can be downloaded by running the update dependencies task. Only applicable for Windows machines
- **java_deps**: java dependencies are installed
- **move_from_temp**: `epitweetr` can atomically move files from temporary folder to data directory
- **geonames**: Geonames.org database is downloaded and indexed
- **languages**: languages vectors are downloaded and indexed
- **geotag**: geotag task has successfully run
- **aggregate**: aggregate task has successfully run
- **alerts**: alerts have been created
- **pandoc**: Pandoc is installed and accessible to epitweetr. Necessary for PDF creation.
- **tex**: a tex distribution is installed and accessible to epitweetr. Necessary for PDF creation.

Management of search and detection loops (Windows)

After activating the search and detection pipeline from the configuration page of `epitweetr` (Windows), two tasks will be created in the task scheduler and two terminal windows will be prompted. Please note that if the computer is logged/turned off or the terminal windows are closed, the search and detection pipeline will stop.

If you activate these tasks from the configuration page of `epitweetr` again, the system will overwrite the tasks created in the task scheduler. Instead, after the first successful activation of these tasks from `epitweetr`, you can easily manage these from the task scheduler. You can stop these tasks by ending and disabling the tasks in the task scheduler, and you can restart these tasks by enabling and running these in the task scheduler.

In the task scheduler, you can establish that the search and detection tasks “run whether the user is logged on or not” to avoid that these tasks stop when you log off or restart the computer.

Management of search and detection loops (Linux and Mac)

Since the search and detection pipeline in Linux or Mac have to be run manually, if the computer is logged/turned off or the terminal windows are closed, the search and detection pipeline will stop. Please remember to follow the steps in the section *Setting up tweet collection and the alert detection loop* to run these tasks again.

Running search and detection pipeline

“Cannot execute task #####: the task is already running”

The detect and the search loop create two files containing their process IDs located in the `epitwitter` data folder: search.PID and detect.PID. This error arises if `epitweetr` finds another R process currently running with the same ID. In order to fix this error you should first verify if the search / detect loop are not already running in another R session. If this is the case, you should not try to start the task since `epitweetr` only supports one instance of the same task running in the same data folder. If the running process is not associated with the task then you can manually delete the PID file and try to start it again.

Change the user of the Twitter authentication when using a Twitter account

1. End and disable the search loop/task in the task scheduler (Windows) or close the R/terminal window with the search loop/task (Linux and Mac)
2. Search for a file called ".rtweet_token" in hidden files. It is usually saved in the Documents folder.
3. Delete that file.
4. Click on "Update properties" in the configuration page of `epitweetr`.
5. Enable and run the search loop/task in the task scheduler (Windows) or run the command in a new R/terminal window with the search loop/task (Linux and Mac). More details are available in the section *"Setting up tweet collection and the alert detection loop"*

Downloading GeoNames and/or languages

"The specified size exceeds the maximum representable size. Error: Could not create the Java Virtual Machine"

If this error appears when running GeoNames, it means that the machine has Java 32bits. You need to install Java 64bits. And make it accessible to epitwitter either by setting "JAVA_HOME" environment variable or by setting the right java binary on the system PATH.

"Max number of retried reached failed while processing languages. Error in get_geolocated_period(dataset): To aggregate, or calculate alerts geolocation must have been successfully executed, but no geolocation files were found"

If you see this error in the configuration page, it means that `epitweetr` could not geotag the collected tweets. It is highly recommended to run GeoNames and languages again since these may have not been properly downloaded. When running this task, you need to ensure that the machine is not logged/turned off or does not go into sleep mode.

The "Launch slots" in the configuration page show NAs instead of the time slots

If it is the first time that you install and launch `epitweetr`, the geotag task of the detection pipeline has to be run at least once in order to see time slots in the "Launch slots" in the configuration page.

Downloading PDF of the dashboard

"Error in: LaTeX failed to compile C:\Users\name~1\...\file######.tex."

This error appears in Windows when clicking on "PDF" in the dashboard and no PDF is saved. The reason is that the path to TEMP and TMP environment variables of the user are too long, Windows shortens the path and epitweetr cannot find this new path. Please follow the next steps to fix this:

1. Open the "environment variable for your account"
2. Change the path for TEMP and TMP to a shorter path (e.g. "C:\Temp"). The same path should be used for both environment variables.
3. Log off and log on
4. You can now download and save the PDF from the dashboard

"Error: pandoc document conversion failed with error 6"

1. Downloading this script (https://raw.githubusercontent.com/jgm/pandoc/master/macos/uninstall-pandoc.pl)

2. Uninstall pandoc (https://pandoc.org/installing.html) by running perl uninstall-pandoc.pl

Different totals in dashboard outputs

When counting the total tweets in the dashboard of the Shiny app or in the downloadable data, you might get differences in the total numbers of tweets between the three outputs. This might be due to the following reasons:

1. World (all) versus World (geolocated)
    - The default option for the regions in World (all), this means that also non-geolocated tweets are included in the trendline, but only geolocated tweets can be visualized in the maps and the most frequent words figure, therefore the overall total of tweets can differ between these outputs wen selecting World (all) or the empty default.
2. Country specific analysis
    - If you select only one country in the filters, the trendline will show all tweets for this country, but the map will show the tweets on a subnational level in the map. It could be that some tweets might have been geolocated to a certain country, but without further subnational data. These tweets will then be visible in the trendline total, but not in the subnational bubbles in the map.
3. Most frequent words
    - In contrast to the other outputs in the dashboard, the most frequent words figure is always based on tweet location regardless of the filter (due to memory capacity). Therefore, if user location or both locations are selected in the location filter, this figure might have a different total then the other two outputs.

Receiving only real-time alerts

This relates to users who have selected topics and/or regions for receiving related alerts in real-time or have selected topics and/or regions for receiving related alerts on a scheduled time span. If, in these cases, you only receive real-time alerts with all topics and regions, it may be that no time slots have been included in the subscribers file from the configuration page. These time slots are used for the scheduled alerts and if no slots are included in the file, alerts from all topics and regions are sent as real-time alerts.

Not receiving email alerts

If you do not receive email alerts and you see an error in `epitweetr` referring to denied login, it means that `epitweetr` could not login to the email account provided in the configuration page. Some of the reasons for that are:

- The server or port included in the configuration page are incorrect
- The attempt of `epitweetr` to login to the email account is being blocked by the server. This can happen with some organisational email accounts. In that case, please contact the IT department of your organisation
- If using Gmail account, you need to allow less secure apps in the settings of your account

References

Allévius, Benjamin, and Michael Höhle. 2017. “Prospective Detection of Outbreaks.” *arXiv:1711.08960 [Stat]*, November. https://arxiv.org/abs/1711.08960.

Chatterjee, Samprit, and Ali S. Hadi. 1988. *Sensitivity Analysis in Linear Regression*. Wiley Series in Probability and Mathematical Statistics. New York: Wiley.

Farrington, C. P., N. J. Andrews, A. D. Beale, and M. A. Catchpole. 1996. “A Statistical Algorithm for the Early Detection of Outbreaks of Infectious Disease.” *Journal of the Royal Statistical Society. Series A (Statistics in Society)* 159 (3): 547. doi:10.2307/2983331.

Fricker, Ronald D., Benjamin L. Hegler, and David A. Dunfee. 2008. “Comparing Syndromic Surveillance Detection Methods: EARS’ Versus a CUSUM-Based Methodology.” *Statistics in Medicine* 27 (17): 3407–29. doi:10.1002/sim.3197.

Noufaily, Angela, Doyo Enki, Paddy Farrington, Paul Garthwaite, Nick Andrews, and Andre Charlett. 2013. “An Improved Algorithm for Outbreak Detection in Multiple Surveillance Systems.” *Online Journal of Public Health Informatics* 5 (1): e148. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3692796/.

Salmon, Maëlle, Dirk Schumacher, and Michael Höhle. 2016. “Monitoring Count Time Series in *R* : Aberration Detection in Public Health Surveillance.” *Journal of Statistical Software* 70 (10). doi:10.18637/jss.v070.i10.