ETACTS Program Development Document and Experiment

Yingcheng Sun
Last updated: 05/31/2020

1. API

eTACTS website scrapes clinical trial information from clinicaltrials.gov website in real-time instead of saving all the data locally, so it uses many APIs offered by clinicaltrials.gov:

1) Search API for the index page:
'http://clinicaltrials.gov/search?term=%s&displayxml=true' % txt ("txt" is what you want to search)
example: https://clinicaltrials.gov/search?term=covid&displayxml=true
This API will return 20 clinical trial results with a few basic information like NCTID, title, condition for each of them.
It is called in:
/app/lib/ctgov.py
by the function `def search (txt, npag)`

2) Advanced Search API for the index page:
'http://clinicaltrials.gov/ct2/results?%sdisplayxml=True' % ctg_param ("txt" is the search condtions)
This API will return 20 clinical trial results with a few basic information like NCTID, title, condition for each of them.
It is called in:
/app/lib/ctgov.py
by the function `def form_advanced_search_url (param)`

3) Search API for the tag cloud page:
First, it uses *url = 'http://clinicaltrials.gov/search?term=%s&displayxml=true' % txt*
To get the total number of clinical trials for the search term (*count = n*), and called in
/app/lib/ctgov.py
By `def get_initial_nct (txt)`
Then it uses
'https://clinicaltrials.gov/search?term= %s&count=n%ddisplayxml=true' % s&count=n
to obtain the list of clinical trial with the basic information for each of them (only NCTID is scraped and used).
Example: https://clinicaltrials.gov/search?term=covid&count=100&displayxml=true
It is called in
/app/lib/ctgov.py
By `def get_initial_nct_from_url (url)`

However, the clinicaltrials. gov website does not allow querying more than 1000 trials each request by the above API, so even you set "count=2000", it will only return 1000 trials.

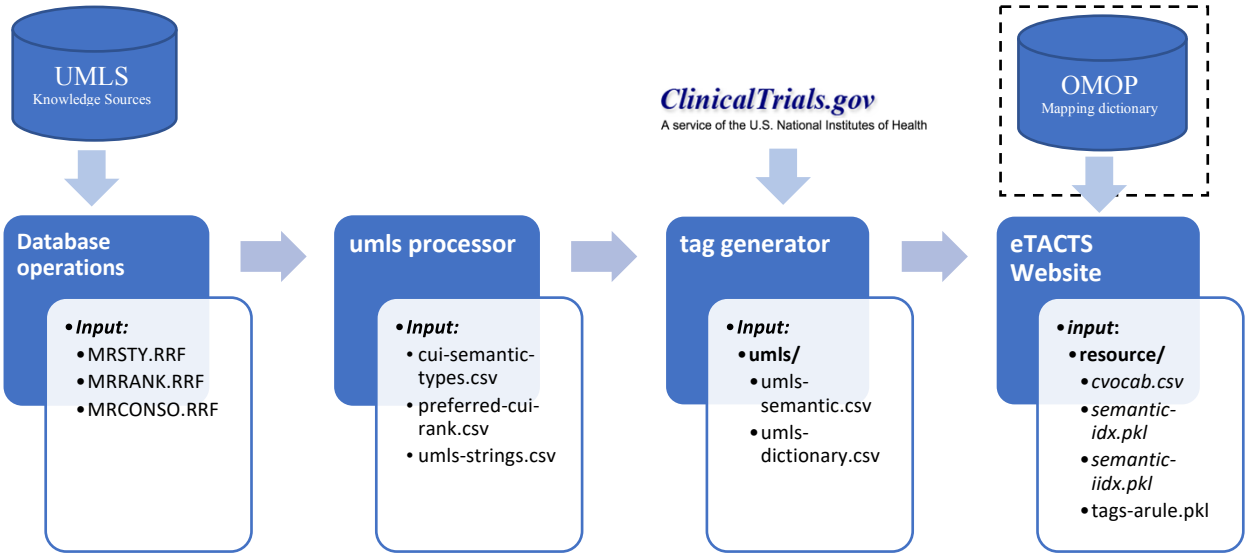
In the updated version, the above API is replaced by a new API without query limitation:

'https://clinicaltrials.gov/api/query/field_values?expr=%s&field=NCTId'% txt
Example: https://clinicaltrials.gov/api/query/field_values?expr=covid&field=NCTId
It does not need the "count=n" expression and will return all the trials one time.
More details on how to use this API can be found in: https://clinicaltrials.gov/api/gui

4) Clinical trial detailed information querying API for the tag cloud page
'http://clinicaltrials.gov/ct2/show/%s?displayxml=true' %nctid
example: https://clinicaltrials.gov/ct2/show/NCT01306084?displayxml=true
It will return all the detailed information given the clinical trial ID, and its title and conditions will be parsed and displayed on the tag cloud page. This querying process will be iteratively repeated for at most 20 times each time to display enough clinical trials on the page .

2. Data Flow

The eTACTS program includes three independent sub-programs: the website (eTACTS), tag generator (nct engine) and the UMLS processor. The data flow is visualized as the picture below shows:



2.1 Database operations

To generate files used by "umls processor", we need to install UMLS Knowledge Sources to your computer first following the instruction in UMLS official website:
https://www.nlm.nih.gov/research/umls/implementation_resources/metamorphosys/help.html

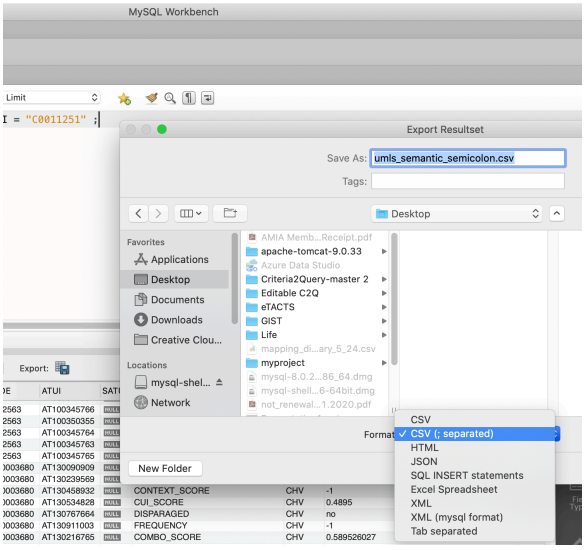
And then install a database, such as MySQL or MSSQL.

Next, create tables needed and load the data to the database. There are a couple of ".RRF" files offered by UMLS, but only three of them are necessary to be loaded to the database: MRSTY.RRF, MRRANK.RRF and MRCONSO.RRF. Examples for MySQL:

```
DROP TABLE IF EXISTS MRSTY;
CREATE TABLE MRSTY (
    CUI char(8) NOT NULL,
    TUI char(4) NOT NULL,
    STN varchar(100) NOT NULL,
    STY varchar(50) NOT NULL,
    ATUI    varchar(11) NOT NULL,
    CVF int unsigned
) CHARACTER SET utf8;

load data local infile 'MRSTY.RRF' into table MRSTY fields terminated by '|' ESCAPED BY '' lines terminated by '\n'
(@cui,@tui,@stn,@sty,@atui,@cvf)
SET CUI = @cui,
TUI = @tui,
STN = @stn,
STY = @sty,
ATUI = @atui,
CVF = NULLIF(@cvf,'');

DROP TABLE IF EXISTS MRXNS_ENG;
CREATE TABLE MRXNS_ENG (
    LAT char(3) NOT NULL,
    NSTR    text NOT NULL,
    CUI char(8) NOT NULL,
    LUI varchar(10) NOT NULL,
    SUI varchar(10) NOT NULL
) CHARACTER SET utf8;
DROP TABLE IF EXISTS MRRANK;
CREATE TABLE MRRANK (
    MRRANK_RANK int unsigned NOT NULL,
    SAB varchar(40) NOT NULL,
    TTY varchar(40) NOT NULL,
    SUPPRESS    char(1) NOT NULL
) CHARACTER SET utf8;

load data local infile 'MRRANK.RRF' into table MRRANK fields terminated by '|' ESCAPED BY '' lines terminated by '\n'
(@mrrank_rank,@sab,@tty,@suppress)
SET MRRANK_RANK = @mrrank_rank,
SAB = @sab,
TTY = @tty,
SUPPRESS = @suppress;
DROP TABLE IF EXISTS MRCONSO;
CREATE TABLE MRCONSO (
    CUI char(8) NOT NULL,
    LAT char(3) NOT NULL,
    TS  char(1) NOT NULL,
    LUI varchar(10) NOT NULL,
    STT varchar(3) NOT NULL,
    SUI varchar(10) NOT NULL,
    ISPREF  char(1) NOT NULL,
    AUI varchar(9) NOT NULL,
    SAUI    varchar(50),
    SCUI    varchar(100),
    SDUI    varchar(100),
    SAB varchar(40) NOT NULL,
    TTY varchar(40) NOT NULL,
    CODE    varchar(100) NOT NULL,
    STR text NOT NULL,
    SRL int unsigned NOT NULL,
```

```
    SUPPRESS    char(1) NOT NULL,
    CVF int unsigned
) CHARACTER SET utf8;

load data local infile 'MRCONSO.RRF' into table MRCONSO fields terminated by '|' ESCAPED BY '' lin
es terminated by '\n'
(@cui,@lat,@ts,@lui,@stt,@sui,@ispref,@aui,@saui,@scui,@sdui,@sab,@tty,@code,@str,@srl,@suppress,@
cvf)
SET CUI = @cui,
LAT = @lat,
TS = @ts,
LUI = @lui,
STT = @stt,
SUI = @sui,
ISPREF = @ispref,
AUI = @aui,
SAUI = NULLIF(@saui,''),
SCUI = NULLIF(@scui,''),
SDUI = NULLIF(@sdui,''),
SAB = @sab,
TTY = @tty,
CODE = @code,
STR = @str,
SRL = @srl,
SUPPRESS = @suppress,
CVF = NULLIF(@cvf,'');
```

Finally, use SQL statement to select data from tables and export to CSV format files. Since there will be comma or quotation mark within the medical terms, it is better to dump the data with CSV files separated by semicolon, as the picture shows.



Exported files and SQL statements
“umls-semantic.csv”: SELECT distinct STY FROM umls.MRSTY ;
“cui-semantic-types.csv”: SELECT CUI,STY FROM umls.MRSTY ;
“umls-strings.csv”: SELECT CUI, STR FROM umls.MRCONSO ;
“preferred-cui-rank.csv”:

SELECT MRCONSO.CUI,MRCONSO.STR,MRRANK.MRRANK_RANK FROM MRRANK, MRCONSO where MRRANK.SAB=MRCONSO.SAB and MRRANK.TTY= MRCONSO.TTY and MRRANK.SUPPRESS = MRCONSO.SUPPRESS;

Reference:
find all information of a UMLS concept
https://www.nlm.nih.gov/research/umls/implementation_resources/query_diagrams/er1.html

Concept Name Ranking (File = MRRANK.RRF)
https://www.ncbi.nlm.nih.gov/books/NBK9685/table/ch03.T.concept_name_ranking_file_mrrank/

Perform a norm string search
https://www.nlm.nih.gov/research/umls/implementation_resources/scripts/README_ORF_MySQL_Output_Stream.html

2.2 UMLS processor

Given the input files "cui-semantic-types.csv", "preferred-cui-rank.csv" and "umls-strings.csv", "umls processor" program will generate the "umls-dictionary.csv" file. Please take care that you need to update the CSV reader function:

```
1.  # read data from a csv file seperated by semicolon and with title
2.  def read_csv (filename, logout = True):
3.      try:
4.          reader = csv.reader (open(filename, "r"), delimiter=';')
5.          data = []
6.          for r in islice(reader, 1, None):
7.              data.append(r)
8.          return data
9.      except Exception as e:
10.         if logout is True:
11.             log.error('%s: %s' % (e, filename))
12.         return None
```

2.3 NCT Engine

The NCT engine needs the following files as input:

negation-rules.txt
stop-words/
    English.csv
    medical.csv
treebank-tags.csv
umls/
    umls-semantic.csv
    umls-dictionary.csv

Update the paths in ”tag-mining.sh”, “indexing.sh” and “arule-mining.sh” first, and then run the “NCT engine” program in the following order:

1) Run ”tag-mining.sh” to generate “cvocab.csv” first. You can modify the “get_clinical_trials” function in “ctgov.py” to create your own clinical trial candidate list.
2) Run “indexing.sh” to generate “nctec-cindex.pkl”, and rename it to “*semantic-idx.pkl*”.
3) Run “arule-mining.sh” to generate “tags-arule.pkl”.

2.4 eTACTS Website

The “resource” folder in eTACTS includes all the data information:
“cvocab.csv”: tag list and its type

“tags-arule.pkl”: tag indexed by types, for example
('inc:hiv infections',), [('inc:hiv', 0.5)]
('exc:pneumothorax',), [('inc:oxygen therapy care', 1.0)]

“*semantic-idx.pkl*”: NCTID and the tags included in this trial, for example:
'NCT04359836': set(['inc:male gender', 'minimum age = 18', 'exc:refusal', 'gender = all'])
'NCT04362943': set(['minimum age = 70', 'gender = all'])

“*semantic-iidx.pkl*”: the transposed matrix of “*semantic-idx.pkl*”, tag and the ID list of clinical trials that contains such tag, for example:

'inc:venous thrombosis': set(['NCT04331613', 'NCT04365127', 'NCT04377997', 'NCT04390217', 'NCT04338126'])
'inc:hemorrhage': set(['NCT04377997', 'NCT04352985']

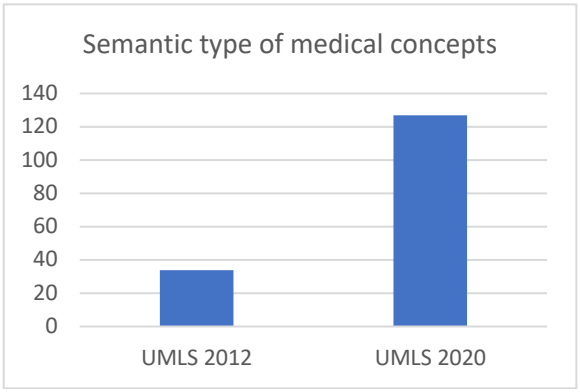
“*semantic-iidx.pkl*” is generated by “*semantic-idx.pkl*” when the eTACTS website is loaded at the first time.

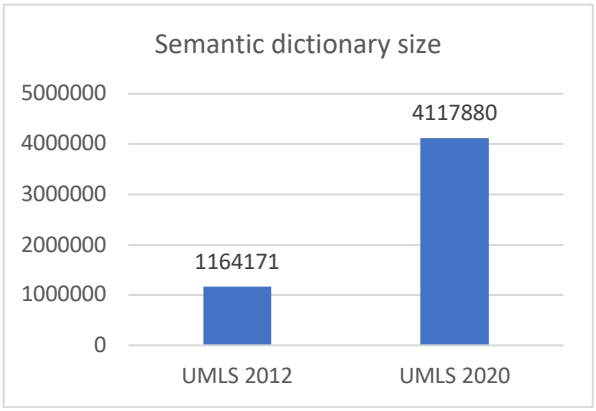
2.5 Use OMOP Database

OMOP Common Data Model might be another option for generating resource files that eTACTS website needs. Given the eligibility criteria corpus scraped from CT.gov and the “OMOP_mapping_dictionary.csv” file, it might be possible to generate the *“cvocab.csv”* and “*semantic-idx.pkl*” files. (all other needed resource files can be derived from these two files)

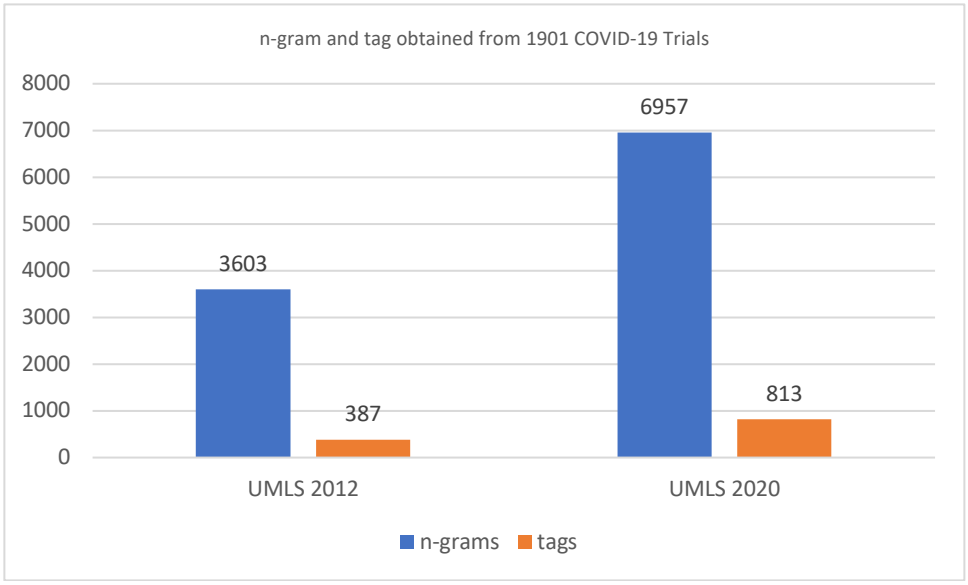
3 Experiment

3.1 Dataset





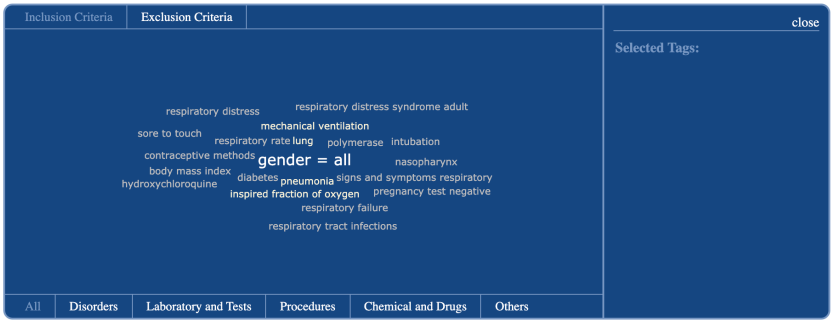
The difference between UMLS 2012 and UMLS 2020 in semantic types of medical concepts and sematic dictionary size are showed in the above pictures. We use n-grams composed by at most 5 words, and select tags based on the probability of frequent tag min frequency value 0.02. We retained 387 tags appearing at least 18 times with UMLS 2012 and 813 tags appearing at least 19 times with UMLS 2020. The results are shown in the below picture.



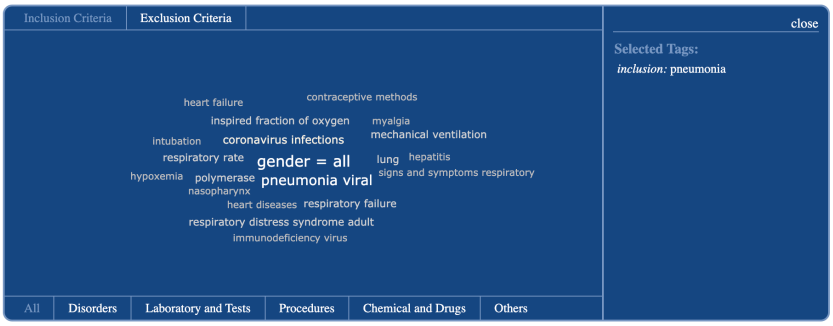
3.2 Tag cloud for COVID-19 related trials

Following the default setting of eTACTS, the tag cloud window size is set as 20 and tag random parameter is set as 10.

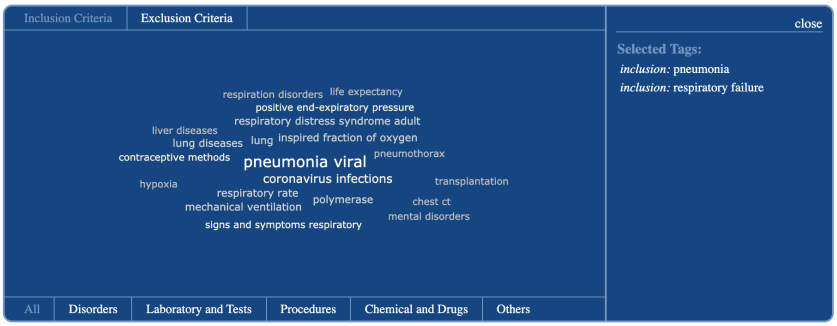
1. Tag cloud generated by UMLS 2012



Found 1,901 clinical trials for: COVID-19

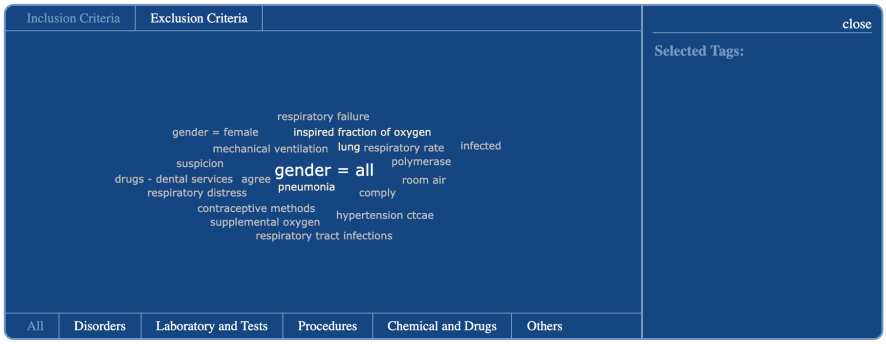


Left 229 clinical trials for: COVID-19

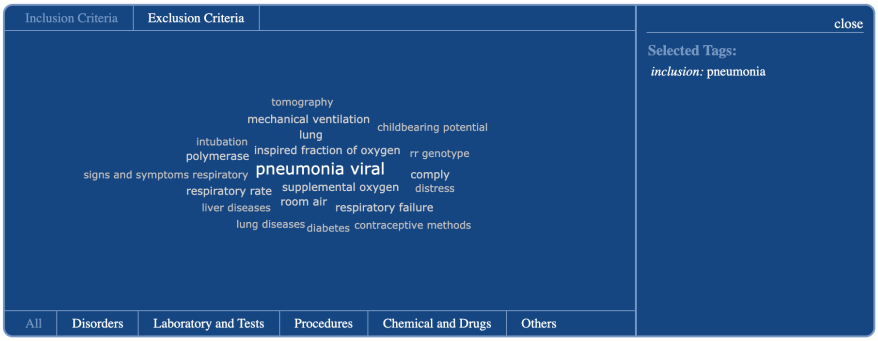


Left 22 clinical trials for: COVID-19

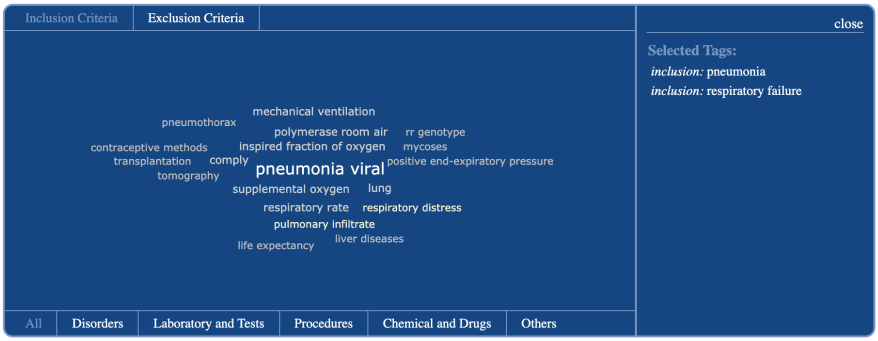
2. Tag cloud generated by UMLS 2020



Found 1,901 clinical trials for: covid-19



Left 227 clinical trials for: covid-19



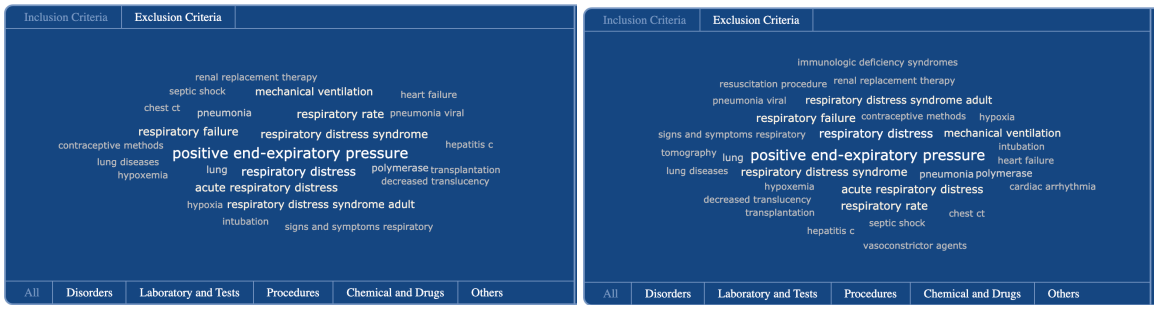
Left 22 clinical trials for: covid-19

3.3 Tag cloud size

We have two global variables in views.py: *tagcloud_size* and *tag_rand.*

Given all tags sorted by their frequency from high to low, top "*tagcloud_size + tag_rand*" number of tags will be selected at first and then their order will be randomized. Finally, the "*tagcloud_size*" number of tags will be selected from the randomized tags. The randomization algorithm is implemented by "*def random_tags (cloud, ntag, nrand)*" in "*tagcloud.py*".

We would like to show users as many tags as possible each time, but if the tag cloud size is more than 25, the tag cloud window will look be a little crowded, so we set *tagcloud_size=25* and *tag_rand=0* to display the most frequent tags.



tagcloud_size=25 vs tagcloud_size=30