COVID19 – Research Literature Clustering

Avinash Shanker
University of Texas, Arlington
Volunteering Under Dr. Elizabeth Diaz

Abstract

Given the large number of research literature and the rapid spread of COVID-19, it is difficult for health professionals to keep up with new information on the virus. So, clustering similar research articles together can simplify the search for related publications. By using clustering for labelling in combination with dimensionality reduction for visualization, the collection of literature can be represented by a scatter plot. On this plot, publications of highly similar topic will share a label and will be plotted near each other. In order, to find meaning in the clusters, topic modelling will be performed to find the keywords of each cluster. By using Bokeh, the plot will be interactive. User's will have the option of seeing the plot as a whole or filtering the data by cluster. If a narrower scope is required, the plot will also have a search function which will limit the output to only papers containing the search term. Hovering over points on the plot will give basic information like title, author, journal, and abstract. Clicking on a point will bring up a menu with a URL that can be used to access the full publication.

The fundamental purpose is to make it easier for trained professionals to sift through many, many publications related to the virus, and find their own determinations using the CORD-19 dataset.

Introduction

For the purpose of this project, I will be using CORD-19 dataset. In response to the COVID-19 pandemic, the White House and a coalition of leading research groups have prepared the COVID19 Open Research Dataset (CORD-19). CORD-19 is a resource of over 200,000 scholarly articles, including over 93,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses. The CORD-19 dataset represents the most extensive machine-readable coronavirus literature collection available for data mining to date.

Approach:

- Parse the text from the body of each document using Natural Language Processing.
- Turn each document instance into a feature vector Xi using Term Frequency–inverse Document Frequency.
- Apply Dimensionality Reduction to each feature vector using t-Distributed Stochastic Neighbor Embedding to cluster similar research articles in the two-dimensional plane.
- Use Principal Component Analysis (PCA) to project down the dimensions of X to a number of dimensions that will keep variance while removing noise and outliers.

• Apply k-means clustering where k value is to be determined by elbow method or Silhouette Algorithm to label each cluster.
• Apply Topic Modeling on X using Latent Dirichlet Allocation (LDA) to discover keywords from each cluster.
• Investigate the clusters visually on the plot, zooming down to specific articles as needed, and via classification using Stochastic Gradient Descent (SGD).

Methodology

Implementation
CORD19 dataset is split into two parts. First, it contains a metadata file which has only the essential details regarding the research publication like unique ID, abstract, published date, Body ID as JSON file.

We need to link each record with its research paper's abstract with its body and make a data frame out of it.

The purpose of creating a dataframe is to get the count of frequent terms in each research paper which can be used for clustering. Dataframe is of structure {paper_id, doi, abstract, body_text, authors, title, journal, abstract_summary}.

During constructing of dataframe we need some additional fields such as word count, unique count.

CORD-19 is a massive dataset it requires preprocessing of data then removing duplicates and removing word like {that, this, and, or, will, the} which add no actual. Duplicates may have been caused because of author submitting the article to multiple journals. Hence, we need to remove such records from dataset.

Since, the CORD-19 dataset contains not just English other languages such as Spanish, French or Chinese. We need to separate such records. And for the purpose of this work, will only be working with English language work. I am importing langdetect from detect library for this purpose.

As, we previously discussed the about preprocessing data which requires removing stop words in research papers which add no value to making clusters. Such values need to be removed else they will account for significant noise. We will also be using spacy library to process the string from punctuations and case sensitiveness.

Next, step is vectorization of the pre-processed data. Now that we have pre-processed the data, it is time to convert it into a format that can be handled by our algorithms. For this purpose, will use dimensionality eeduction to each feature vector using t-Distributed Stochastic Neighbor. This will convert our string

formatted data into a measure of how important each word is to the instance out of the literature as a whole.

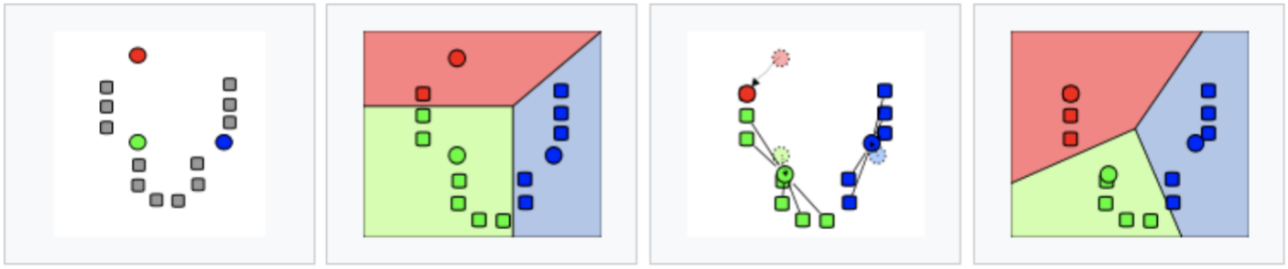
Vectorization

Now that we have pre-processed the data, it is time to convert it into a format that can be handled by our algorithms. For this purpose we will be using tf-idf. This will convert our string formatted data into a measure of how important each word is to the instance out of the literature as a whole.

PCA & Clustering

Let's see how much we can reduce the dimensions while still keeping 95% variance. We will apply Principle Component Analysis (PCA) to our vectorized data. The reason for this is that by keeping a large number of dimensions with PCA, you don't destroy much of the information, but hopefully will remove some noise/outliers from the data, and make the clustering problem easier for k-means. Note that X_reduced will only be used for k-means, t-SNE will still use the original feature vector X that was generated through tf-idf on the NLP processed text.

To separate the literature, k-means will be run on the vectorized text. Given the number of clusters, k, k-means will categorize each vector by taking the mean distance to a randomly initialized centroid. The centroids are updated iteratively.



1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).

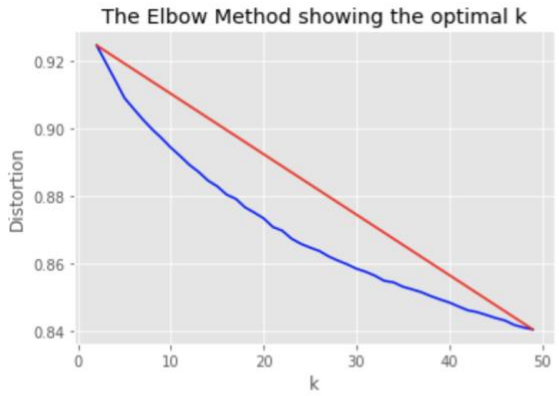
2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3. The centroid of each of the *k* clusters becomes the new mean.

4. Steps 2 and 3 are repeated until convergence has been reached.

How many clusters

To find the best k value for k-means we'll look at the distortion at different k values. Distortion computes the sum of squared distances from each point to its assigned center. When distortion is plotted against k there will be a k value after which decreases in distortion are minimal. This is the desired number of clusters.

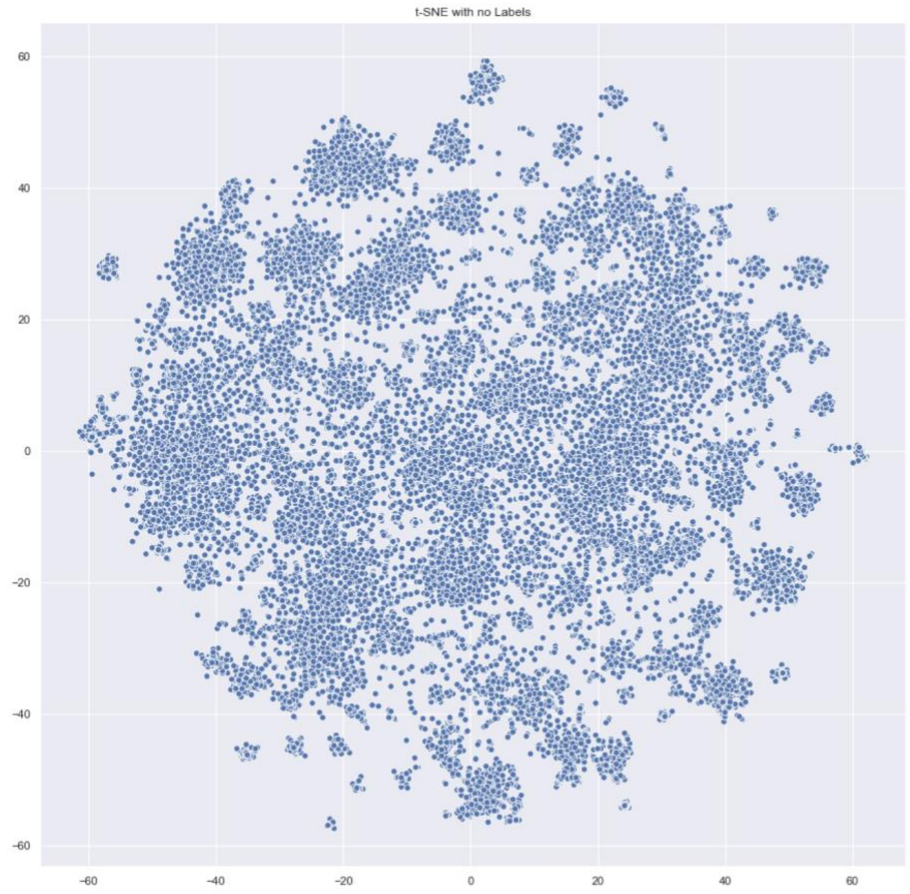


In this plot we can see that the better k values are between 18-25. After that, the decrease in distortion is not as significant. For simplicity, we will use k=20

Dimensionality Reduction with t-SNE

Using t-SNE we can reduce our high dimensional features vector to 2 dimensions. By using the 2 dimensions as x,y coordinates, the body_text can be plotted.
t-Distributed Stochastic Neighbor Embedding (t-SNE) reduces dimensionality while trying to keep similar instances close and dissimilar instances apart. It is mostly used for visualization, in particular to visualize clusters of instances in high-dimensional space

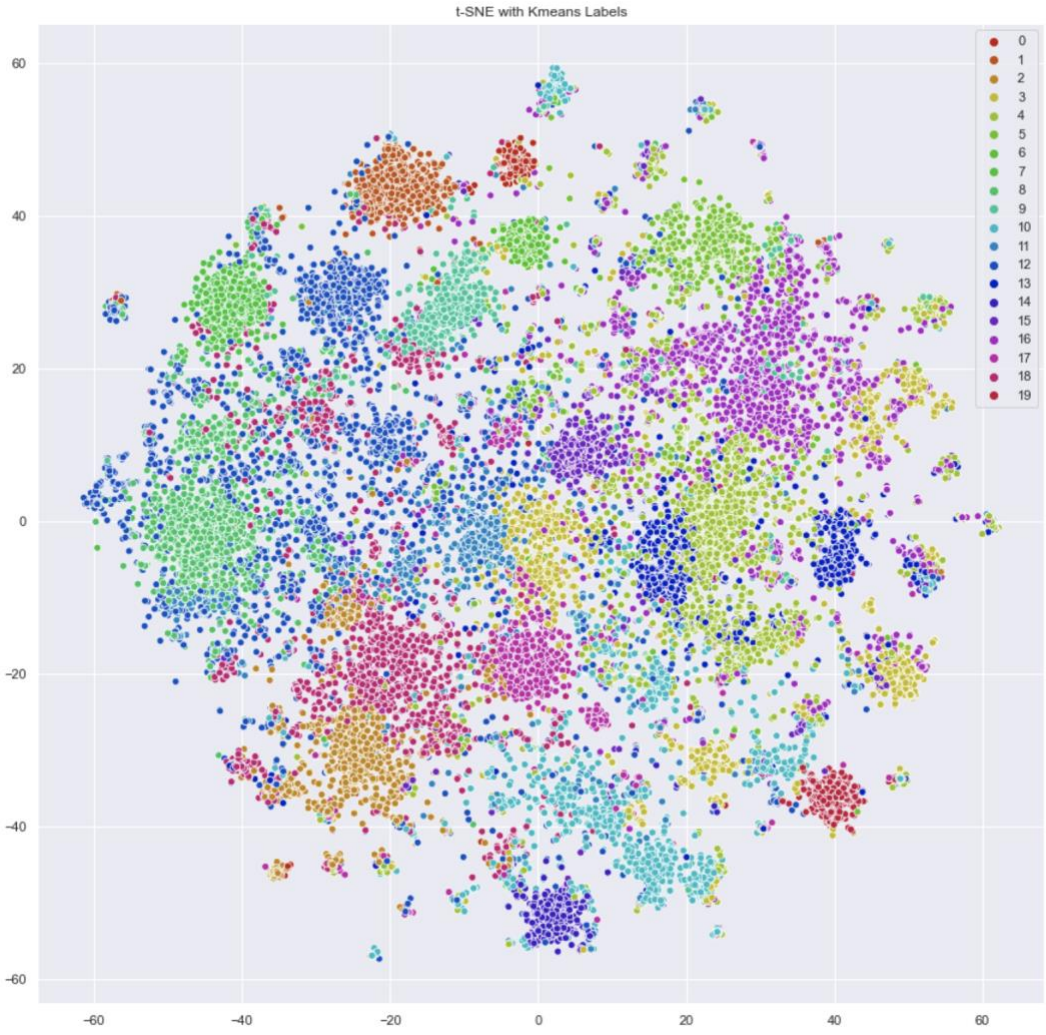
The labeled plot gives better insight into how the papers are grouped. It is interesting that both k-means and t-SNE are able to agree on certain clusters even though they were ran independently. The location of each paper on the plot was determined by t-SNE while the label (color) was determined by k-means. If we look at a particular part of the plot where t-SNE has grouped many articles forming a cluster, it is likely that k-means is uniform in the labeling of this cluster (most of the cluster is the same color). This behavior shows that structure within the literature can be observed and measured to some extent.



This looks pretty bland. There are some clusters we can immediately detect, but the many instances closer to the center are harder to separate. t-SNE did a good job at reducing the dimensionality, but now we need some labels. Let's use the clusters found by k-means as labels. This will help visually separate different concentrations of topics.

Now there are other cases where the colored labels (k-means) are spread out on the plot (t-SNE). This is a result of t-SNE and k-means finding different connections in the higher dimensional data. The topics of these papers often intersect so it hard to cleanly separate them. This effect can be observed in the formation of subclusters on the plot. These subclusters are a conglomeration of different k-means labels but may share some connection determined by t-SNE.

This organization of the data does not act as a simple search engine. The clustering + dimensionality reduction is performed on the mathematical similarities of the publications. As an unsupervised approach, the algorithms may even find connections that were unapparent to humans. This may highlight hidden shared information and advance further research.



Topic Modeling on Each Cluster

Now we will attempt to find the most significant words in each cluster. K-means clustered the articles but did not label the topics. Through topic modeling we will find out what the most important terms for each cluster are. This will add more meaning to the cluster by giving keywords to quickly identify the themes of the cluster.
For topic modeling, we will use LDA (Latent Dirichlet Allocation). In LDA, each document can be described by a distribution of topics and each topic can be described by a distribution of words.

Classify

Though arbitrary, after running kmeans, the data is now 'labeled'. This means that we now use supervised learning to see how well the clustering generalizes. This is just one way to evaluate the clustering. If k-

means was able to find a meaningful split in the data, it should be possible to train a classifier to predict which cluster a given instance should belong to.

Plotting the data¶

The previous steps have given us clustering labels and a dataset of papers reduced to two dimensions. By pairing this with Bokeh, we can create an interactive plot of the literature. This should organize the papers such that related publications are in close proximity. To try to understand what the similarities may be, we have also performed topic modelling on each cluster of papers in order to pick out the key terms.

Bokeh will pair the actual papers with their positions on the t-SNE plot. Through this approach it will be easier to see how papers fit together, allowing for both exploration of the dataset and evaluation of the clustering.

Conclusion

In this project, we have attempted to cluster published literature on COVID-19 and reduce the dimensionality of the dataset for visualization purposes. This has allowed for an interactive scatter plot of papers related to COVID-19, in which material of similar theme is grouped together. Grouping the literature in this way allows for professionals to quickly find material related to a central topic. Instead of having to manually search for related work, every publication is connected to a larger topic cluster. The clustering of the data was done through k-means on a pre-processed, vectorized version of the literature's body text. As k-means simply split the data into clusters, topic modeling through LDA was performed to identify keywords. This gave the topics that were prevalent in each of the clusters. Both the clusters and keywords are found through unsupervised learning models and can be useful in revealing patterns that humans may not have even thought about. In no part of this project did we have to manually organize the papers: the results are due to latent connections in the data.

K-means (represented by colors) and t-SNE (represented by points) were able to independently find clusters, showing that relationships between papers can be identified and measured. Papers written on highly similar topics are typically near each other on the plot and bear the same k-means label. However, due to the complexity of the dataset, k-means and t-SNE will sometimes arrive at different decisions. The topics of much of the given literature are continuous and will not have a concrete decision boundary. This may mean that k-means and t-SNE can find different similarities to group the papers by. In these conditions, our approach performs quite well.

As this is an unsupervised learning problem, the evaluation of our work was not an exact science. First, the plot was examined to assert that clusters were actually being formed. After being convinced of this, we examined the titles/abstracts of some of the papers in different clusters. For the most part, similar research areas were clustered. Our last evaluation method was classification. By training a classification model with the k-means labels and then testing it on a separate subset of the data, we could see that the clustering was not completely arbitrary as the classifier performed well.

Our manual inspection of the documents was quite limited, as neither of the authors are qualified to assess the meaning of the literature. Even so, it was apparent that articles on key topics could be easily found in close proximity to each other. For example, searching for 'mask' can reveal a sub cluster of papers that evaluate the efficacy of masks. We believe that health professionals can use this tool to find real links in the texts. By organizing the literature, qualified people can quickly find related publications that answer the task questions. This project can further be improved by abstracting the underlying data analysis techniques as described in this notebook to develop a user interface/tool that presents the related articles in a user-friendly manner.

Future thoughts:

- Possible false positives, difficult to draw an exact line between subjects
- K-means and t-SNE are unsupervised approaches that will not necessarily group instances in a predictable way. Due to their unsupervised nature, there is no 'right answer' for how the papers should be clustered. This could be difficult to debug if problems arise.
- Loss of foreign language papers. This leads to the loss of experience from different geographic locations on dealing with COVID-19
- The algorithms used in this notebook are stochastic so the results may vary depending on the random state. In this notebook all of the algorithms are set to random state 42 (the meaning of life) to ensure reproducible results
- Long run time to train models on large dataset of literature

Link to the project on my GitHub page:
https://github.com/avinash273/COVID19-Literature-Clustering/blob/master/COVID19_Clustering_Final.ipynb

Few Snapshots of code

jupyter COVID19_literature_clustering_version1 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

Loading the Data

Load the data following the notebook by Ivan Ega Pratama, from Kaggle.

Cite: Dataset Parsing Code | Kaggle, COVID EDA: Initial Exploration Tool

Loading Metadata

In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import glob
import json
import matplotlib.pyplot as plt
```

Let's load the metadata of the dateset. 'title' and 'journal' attributes may be useful later when we cluster the articles to see what kinds of articles cluster together.

In [2]:
```
root_path = '/Users/avinashshanker/Desktop/UTAMasters/DataMining/COVID19-Literature-Clustering-master/data/'
metadata_path = f'{root_path}/metadata.csv'
plt.style.use('ggplot')
meta_df = pd.read_csv(metadata_path, dtype={
    'pubmed_id': str,
    'Microsoft Academic Paper ID': str,
    'doi': str
})
meta_df.head()
```

```
/Users/avinashshanker/opt/anaconda3/lib/python3.7/site-packages/IPython/core/interactiveshell.py:3058: DtypeWarning:
Columns (1,5,13,14,15,16) have mixed types. Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

Out[2]:

	cord_uid	sha	source_x	title	doi	pmcid	pubmed_id	license	abstract	publish_time
0	ug7v899j	d1aafb70c066a2068b02786f8929fd9c900897fb	PMC	Clinical features of culture-proven Mycoplasma...	10.1186/1471-2334-1-6	PMC35282	11472636	no-cc	OBJECTIVE: This retrospective chart review des...	2001-07-04

Load the Data into DataFrame ¶

```
Using the helper functions, let's read in the articles into a DataFrame that can be used easily:
```

In [7]:

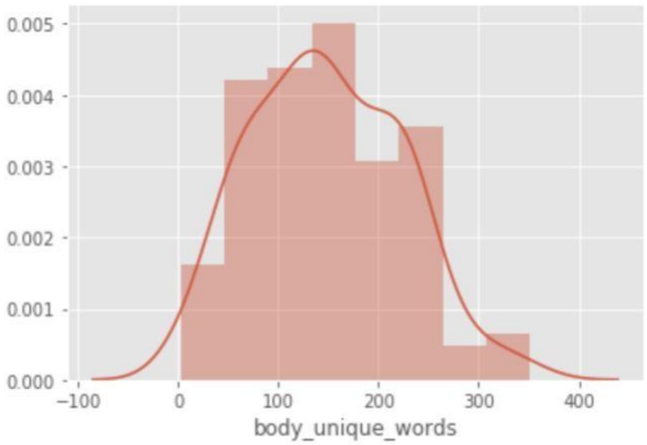
```
dict_ = {'paper_id': [], 'doi':[], 'abstract': [], 'body_text': [], 'authors': [], 'title': [], 'journal': [], 'abs
for idx, entry in enumerate(all_json):
    if idx % (len(all_json) // 10) == 0:
        print(f'Processing index: {idx} of {len(all_json)}')

    try:
        content = FileReader(entry)
    except Exception as e:
        continue  # invalid paper format, skip

    # get metadata information
    meta_data = meta_df.loc[meta_df['sha'] == content.paper_id]
    # no metadata, skip this paper
    if len(meta_data) == 0:
        continue

    dict_['abstract'].append(content.abstract)
    dict_['paper_id'].append(content.paper_id)
    dict_['body_text'].append(content.body_text)

    # also create a column for the summary of abstract to be used in a plot
    if len(content.abstract) == 0:
        # no abstract provided
        dict_['abstract_summary'].append("Not provided.")
    elif len(content.abstract.split(' ')) > 100:
        # abstract provided is too long for plot, take first 100 words append with ...
        info = content.abstract.split(' ')[:100]
        summary = get_breaks(' '.join(info), 40)
        dict_['abstract_summary'].append(summary + "...")
    else:
        # abstract is short enough
        summary = get_breaks(content.abstract, 40)
        dict ['abstract summary'].append(summary)
```

In [29]:

```
sns.distplot(df['body_unique_words'])
df['body_unique_words'].describe()
```

Out[29]:

```
count    142.000000
mean     148.478873
std       74.590106
min        3.000000
25%       88.250000
50%      144.500000
75%      210.000000
max      351.000000
Name: body_unique_words, dtype: float64
```



Vectorization

Now that we have pre-processed the data, it is time to convert it into a format that can be handled by our algorithms. For this purpose we will be using tf-idf. This will convert our string formatted data into a measure of how important each word is to the instance out of the literature as a whole.

In [30]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
def vectorize(text, maxx_features):

    vectorizer = TfidfVectorizer(max_features=maxx_features)
    X = vectorizer.fit_transform(text)
    return X
```

Vectorize our data. We will be clustering based off the content of the body text. The maximum number of features will be limited. Only the top 2 ** 12 features will be used, eseentially acting as a noise filter. Additionally, more features cause painfully long runtimes.

In [31]:
```
text = df['processed_text'].values
X = vectorize(text, 2 ** 12)
X.shape
```

Out[31]: (142, 4096)

Next steps.. PCA & Clustering

1. Next we need to see how much we can reduce the dimensions while still keeping high variance.
2. We will apply Principle Component Analysis (PCA) to our vectorized data. The reason for this is that by keeping a large number of dimensions with PCA, you don't destroy much of the information, but hopefully will remove some noise/outliers from the data, and make the clustering problem easier for k-means.

In [70]:
```
Image(filename='demo/selected_paper.png', width=600, height=100)
```

Out[70]:

A six-year study on respiratory viral infections in a bull testing facility

Authors: Hägglund, S.. Hjort, M.. Graham, D.A.. Öhagen, P.. Törnquist, M.. Alenius, S.
Link: http://doi.org/10.1016/j.tvjl.2006.02.010

Examples

The plot can be used to quickly find many publications on a similar topic. For example, let's say we're interested in finding information on masks and their effectiveness. To do this we can either try to find a cluster with a keyword such as "mask" or we can search for the term directly and try to identify which cluster it most closely relates to.

If you search for "mask" cluster #12 is well represented. It would be wise to start the looking there. A quick scan of the titles indicates that these publications mainly address different masks and their uses for healthcare workers.

Searching for a single key term may cause you to inadvertenly filter out highly similar papers that use different phrasing. After an initial examination, clear the search term and explore the whole cluster by adjusting the slider. In this case, we would set the slider to 12 and look for more interesting papers in the entire cluster.

Now we will explore a couple task-oriented questions.

What do we know about diagnostics and surveillance?

Diagnostics

- To start, we first searched one of the keyterms from this question ("diagnostic")
- For "diagnostic", cluster #17 stood out immediately
- Next we removed the search term and adjusted the slider to 17
- Inside cluster 17 there exists a main cluster and a smaller, denser sub-cluster just off to the right of the main one. Both this main cluster and the sub-cluster seem to contain useful information on diagnosing viral infections. Just quickly skimming through the publications, we saw papers such as:
  - Evaluation of fast-track diagnostics and TaqMan array card real-time PCR assays for the detection of respiratory pathogens
  - Advances in the diagnosis of respiratory virus infections