

COVID-19 DOCUMENT CLUSTERING

Submitted in partial fulfilment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

ADINI HEMANJALI (37110016),

RITIKA CATHERINE. G (37110645)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119**

MARCH – 2021



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **ADINI HEMANJALI (Reg. No. 37110016)** and **RITIKA CATHERINE. G (Reg. No. 37110645)** who carried out the project entitled "**COVID-19 DOCUMENT CLUSTERING**" under my supervision from **November 2020** to **March 2021**.

Internal Guide

Dr. S. Revathy ,M.E.,Ph.d.,

Head of the Department

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I, **RITIKA CATHERINE. G**, hereby declare that the Project Report entitled “**COVID-19 DOCUMENT CLUSTERING**” is done by me under the guidance of **Dr. S. Revathy ,M.E.,Ph.d.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. S. Revathy ,M.E.,Ph.d.,, Associate Professor**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Coronavirus is a worldwide pandemic, which started in Wuhan, China, in December, has extended to contact every edge of the globe. Many individuals around the planet have been sick from the virus, and countless others have passed away. On January 11, 2020, China reported the first case of COVID death. Before long, the infection had begun to spread all around the world. There are about 116 million cases, and the population is yet to be treated. The study and progress in the research of this virus have reduced researchers' ability to get relevant information over the internet. As mutations of infections and microbes are inescapable, a danger for such pandemics remains. To guarantee that the harm is done is lesser, a literature clustering technique can be used to group relevant information together, making accessibility easier for researchers and enthusiasts alike. Furthermore, by using k-means clustering for labelling the clusters and Principal Component Analysis for dimensionality reduction, t-SNE for better visualization, a scatter plot may be used to represent the grouped collection of literature. On this bokeh plot, articles that are similar share a tag, whereas articles that are not similar are placed so far apart. Latent Dirichlet Allocation will be used to extract keywords from each cluster in order to find credibility in the groups.

TABLE OF CONTENT

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	v
	LIST OF FIGURES	vi
	LIST OF ABBREVIATIONS	vii
1	INTRODUCTION	1
	1.1 EXISTING SYSTEM	2
	1.2 PROPOSED SYSTEM	3
2	LITERATURE SURVEY	4
3	METHODOLOGY	
	3.1 AIM OF PROJECT	9
	3.2 SYSTEM REQUIREMENTS	9
	3.2.1 SOFTWARE REQUIREMENTS	9
	3.2.2 HARDWARE REQUIREMENTS	9
	3.3 OVERVIEW OF THE PLATFORM	9
	3.3.1 PYTHON	9
	3.3.2 MACHINE LEARNING USING PYTHON	12
4	MODULE DESCRIPTION	17
	4.1 SYSTEM ARCHITECTURE	17
	4.2 MODULE DESCRIPTION	17
5	RESULT AND DISCUSSION	22
6	CONCLUSION	24
	REFERENCES	25
	APPENDIX	
	A.SOURCE CODE	38

B.SCREEN SHOTS
C.JOURNAL PAPER

40

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
3.1	DISTINGUISHING OF CLUSTERS	13
3.2	DENSITY- BASED SPATIAL CLUSTERING	13
3.3	CLUSTERING PROCESS	15
3.4	K-MEANS ALGORITHM PARTITION AND OBSERVATIONS INTO K CLUSTERS	16
4.1	SYSTEMARCHITECTURE	17
4.2	ELBOW METHOD	20
4.3	COUNT OF UNIQUE WORDS IN THE BODY TEXT	20
4.4	DISTRIBUTION OF LANGUAGES IN THE DATASET	21
4.5	ELBOW METHOD	26
4.6	T-SNE WITHOUT LABELS	29
4.7	T-SNE WITH K MEANS LABELS	30
5.1	BOKEH PLOT	41
5.2	SEARCH KEY WORD	42
5.3	DESCRIPTION	43
5.4	SLIDER	43

LIST OF ABBREVIATIONS

DOC	DOCUMENT
ML	MACHINE LEARNING
OBSN	OBSERVATION
WHO	WORLD HEALTH ORGANIZATION

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In late December 2019, China health professionals reported cases of pneumonia in Wuhan. A few days later, a study on the history and origin of the novel coronavirus was released and made public for health professionals. SARS-CoV-2 was the name given to the new coronavirus by the researchers. The Corona virus family includes SARS-CoV-2 and other related diseases. Since the latest strain of the virus was discovered, it has gone global, killing millions of people in just couple of days. Coronaviruses are a family of different that has been existed for around three decades, and they cause respiratory illness. A novel coronavirus (novel CoV) is a new virus mutant that was not found identified in the human race, and they have been termed as "COVID-19 virus". By that time, over 118k cases had been reported in 114 countries, with over 4k fatalities. The WHO European Zone had become the pandemic's central focus by mid-March 2020, documenting over 40 percent of all known cases globally. Starting on April 28, 63 percent of the worldwide mortality from the infection was from the Region. Since the primary cases were accounted for, WHO has worked nonstop to help nations plan and react to the COVID-19 pandemic. The White House, in association with leading research groups, has published the CORD-19 dataset, which features papers in an unstructured manner, in reaction to the disease outbreak. SARS-CoV-2 and other coronaviruses are featured prominently in the reports. Over 400k scholarly publications are included in the CORD-19 dataset, with over 150k of them having complete transcript. It is permissible to use natural language processing algorithms and other machine learning techniques to gather insights that will aid in the ongoing struggle against the imminent infection. Machine Learning is a branch of Artificial Intelligence that self-learns by trial and error. If a computer program's success at tasks in T , as measured by P , increases with experience E , it is said to learn from experience E about certain class of tasks T and performance measure P , as Tom Mitchel once said. Machine Learning performs tasks without being told to do so. They have two objectives: to classify data on models and predict possible outcomes using these models. Probability plays a major part in machine learning in terms of decisions and

inferences from samples or populations. There are many sub-fields within machine learning to study different types of data. One of those subfields is Natural Language Processing (NLP). NLP can read, analyse, and manipulate data in general English language (understandable). NLP analyses these English text documents improves them by turning the unstructured data into structured data. Some of the most common NLP algorithms used are Word clouds, topic modelling, principal component analysis (used for dimensionality reduction), etc.

1.2 EXISTING SYSTEM

We are currently standing at the shoreline of the big sea of data that is the information age. And while this era's most precious resource is intellectual property and data there seems to be no bound to how much can be gathered. The big problem is how to interpret or analyse what is sampled because the vast amounts surpass what can be done by humans alone. We need powerful techniques to help us in this endeavour. The study of the much smaller problem only relating to text has spawned multiple fields in academia and industry such as computational linguistics, text data mining, information retrieval, and news analytics, etc.

This is the perfect analogy of the field of textual processing as it is today. Current methods have no concept of true semantics, the meaning of what it analyses, it does not understand words more than the symbols they are made of. Although it is debatable whether true meaning even exists and whether it is achievable by machines or whether it is just the resulting feeling of finding a match in the huge archives that is the human memory, this is better done elsewhere

When processing text a lot of it is removed by shifting away useless rock to find the glittering bits. we have no concept of meaning except in very primitive cases our most sophisticated methods of today rely on single words and possibly their close neighbors while their most important information carriers, structure, grammar, and semantics are mostly ignored. Even though there exist very good grammatical parsers, they can at most reconstruct syntactical structures to a degree.

When trying to understand any unknown data one of the most basic instincts for humans is to look for patterns or structure. The leading question is "What do these points have in common? This leads us to our main topic of this report, grouping or

unsupervised categorization of textual data also known as document clustering. Document clustering is an automatic clustering operation of text documents so that similar or related documents are presented in same cluster, dissimilar or unrelated documents are presented in different clusters [1]. Clustering is an unsupervised learning method which does not need any training step; pre-defined categories and labelled documents. So, there is no need for a training set while applying the clustering algorithms. It just uses the input data in order to find regularities in it. It narrows the search space because one is studying nearly structured data instead of a porridge random points. It can be thought of as the analogy of the “lets plot and see what happens” method possible only when sampling data in a low dimensional space.

1.3 PROPOSED SYSTEM

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.

CHAPTER 2

LITERATURE SURVEY

Document clustering is the task of arranging a set of documents into clusters so that intra cluster documents are similar to each other than inter cluster documents. There are two common clustering algorithms. 1) Partitioning algorithms in which clusters are computed directly. Clustering is done by iteratively swapping objects or groups of objects between the clusters. 2) The hierarchical based algorithms in which a hierarchy of clusters is build. Every cluster is subdivided into child clusters, which form a partition of their parent cluster. Different clustering algorithms produce different results with different features. Hierarchical algorithms are slower than partitioning algorithms but they give better accuracy. Therefore, a clustering algorithm should be chosen based on the applications, as the desirable features are application dependent.

The author of [1] has compared kmeans algorithm with his proposed algorithm and proved that his algorithm is much better than k means because it shows better results on large number of documents and also resolved zero clustering issue. The author of [2] has proposed a novel k – nearest neighbour algorithm As the name suggests the method is dependent on the parameter „k“ which can drastically change the output as we vary its values. When the training set contains classes of unequal sizes, the test data is likely to get classified to a class which has more samples than the actual class it belongs to, if that actual class has less number of samples. The parameter k in this method depends on the size of the smallest class sample. The proposed algorithm tries to take care of this limitation by first considering the size of the smallest class and then selecting the k nearest neighbors

The author of [3] has used term frequency –inverse document frequency weighting scheme. Here weighting scheme plays very important role as it helps in removing number of words which are less important leading to the dimensionality reduction.

According to the author of [4], the disadvantage in k-means algorithm is that, the accuracy and efficiency is varied with the choice of initial clustering centers on choosing it randomly. So in his paper, less similarity based clustering method is

proposed for finding the better initial centroids and to provide an efficient way of assigning the data points to suitable clusters with reduced time complexity.

Existing algorithms for k-means clustering are slow and do not scale to large number of data points and converge to different local minima based on the initializations. A fast greedy k-means algorithm can attack both these drawbacks, but it is a limitation when the algorithm is used for large number of data points, So the author of [5] introduced an efficient method to compute the distortion for this algorithm and his experiment results show that the fast greedy algorithm is superior to other methods and can help users to find the relevant documents more easily than by relevance ranking.

The author of [8][9] has introduced EART (extraction of association rules from text) technique , which uses term frequency-inverse document frequency as weighting scheme. It helps in removing the words which are of least importance thereby leading to important rules which helps in knowledge discovery.

The author of [10] has used Association rules discovery techniques to compare the student's performance in the subjects common at Graduation and Post Graduation level and will predict the factors which can explain their success or failure. The mined association rules reveal various factors like student's interest, curriculum design; teaching and assessment methodologies that can affect students who have failed to attain a satisfactory level of performance in the Post Graduation level.

A new semantic-based model that analyses documents based on their meaning is introduced by the author of [11]. He proposed the model that analyses terms and their corresponding synonyms and/or hypernyms on the sentence and document levels. In this model, if two documents contain different words and these words are semantically related, the proposed model can measure the semantic-based similarity between the two documents. The similarity between documents relies on a new semantic-based similarity measure which is applied to the matching concepts between documents. The author of [12] has discussed the structural and semantic similarity of various XML DTDs for efficient clustering mechanism. As well as by analysing the structure of DTDs, the nested and repeated nodes in the tree has been eliminated in the pre-processing mode and efficient clusters has been made based on similarities of XML trees. This study proves group of structurally similar

XML documents provides most precise results in large document collections and described an efficient clustering strategy which group most similar DTDs where preliminary evaluation made with the synthetic data.

There are several clustering approaches. These are partitioning (eg. K-means, kmedoids), hierarchical (eg. DIANA, AGNES, BIRCH), density-based (eg. DBSACN, OPTICS), grid-based (eg. STING, CLIQUE), model based (eg. EM, COBWEB), frequent pattern-based (eg. p-Cluster), user-guided or constraint-based (eg. COD), and link-based (eg. SimRank, LinkClus) clustering approaches [1]. Most of these are explained and some of them firstly proposed in the book of Kaufman and Rousseeuw in 1990 which are partitioning, hierarchical and fuzzy clustering approaches [2].

The most frequent method which is applied to documents is hierarchical clustering method. In 1988, Willett applied agglomerative clustering methods to documents by changing the calculation method of distance between clusters [3]. These algorithms have several problems with clusters that finding stopping point is very difficult and they run too slowly for thousands of documents. Hierarchical clustering algorithms are applied to documents for several times by Zhao and Karypis [4,5] and in 2005 they tried to improve agglomerative clustering algorithm by adding constrains [6].

K-means and its variants, which are partitioning clustering algorithms that create a non-hierarchical clustering consisting of k clusters, are applied to documents [2]. These algorithms are more efficient and scalable, and their complexity is linear to the number of documents. A disadvantage of k-means is that estimating the value of k wrongly leads worse accuracy. Moreover, k-means can stuck on a local maximum because of randomly chosen initial centroids. In order to solve this problem, Kaufman and Rousseeuw proposed k-medoids 4 algorithm but this algorithm is computationally much more expensive and does not scale well large document sets [2].

A comparison of document clustering techniques is done by Steinbach and et al. in 2000 [7]. In their study, they applied k-means, its variant bisecting k-means, and hierarchical clustering algorithms to documents. It shows that average-link algorithm generally performs better than single-link and complete-link algorithms among hierarchical clustering methods for the document data sets used in the experiments.

On the other hand, average-link algorithm is compared with k-means and bisecting k-means and it has been concluded that bisecting k-means performs better than average-link agglomerative hierarchical clustering algorithm and k-means algorithm in most cases for the data sets used in the experiments. The most recent study on document clustering is done by Liu and Xiong in 2011 [8]. They introduce common text clustering algorithms which are hierarchical clustering, partitioned clustering, density based algorithm and self-organizing maps algorithm, analyse and compare some aspects of clustering algorithms such as the applicable scope, the initial parameters, termination conditions and noise sensitivity.

K.M. Hammouda et al. [2] presents two essential elements of effective document clustering. The document index graph is a novel phrase-based document index model that allows for progressive creation of a phrase-based index of the document set with a focus on potency rather than relying solely on single-term indices. The incremental document clustering algorithmic programme in the second half supports optimising cluster tightness by meticulously observing the pair-wise document similarity distribution inside clusters. When those two sections are merged, an associate is created in the underlying model for reliable and accurate document similarity calculation, resulting in significantly improved web document clustering results.

Sen Xu et al. [10] introduces a spectral strategy for fixing disputes with paper cluster ensembles. Since spectral clustering must eventually decipher a matrix's eigenvalues and eigenvectors, it's computationally untamed for large text datasets. They obtain a feasible algorithmic programme by applying algebraic transformation to the similarity matrix.

Sun Park et al. [9] explain how a database clustering approach based on weighted semantic features and cluster similarity is used to cluster meaningful topics from a document collection. The proposed approach would boost the quality of document clustering by avoiding clustering documents with high topic similarities but no context between cluster and document using weighted semantic features. Furthermore, it employs cluster similarity to eliminate the differences between documents in clusters and prevent NMF's biased inherent

semantics expressed in clusters. The proposed method outclasses other document clustering methods, according to the findings of the experiments

Muhammad Rafi et al. [3] explains the suffix tree data model was helped by comparing and introducing distinct approaches to document clustering. The first would be Efficient Phrase-based Document Clustering, which extracts phrases from documents to construct compact document illustrations and clusters the documents using a similarity-based common suffix tree. The second approach employs a common phrase, which is a sequence-based document clustering method that extracts the document's popular sequence of words. It performs the compact illustration using the regular pattern signified array, and then clusters the compressed documents using the document clustering approach. These algorithms use collective hierarchic document clustering to perform the specific clustering step; the differences between these techniques are the predominant supported phrase abstraction, model illustration as a compact document, and the similarity measures used for clustering.

S. Kumar et al. [8] explains using equivalence relations and fuzzy hierarchical clustering, an algorithm and experimental results for fuzzy clustering of web documents are presented.

R. Agrawal et al. [6] have created a learning method that constructs several clusters for every unknown text dataset and correctly clusters the files based on Cosine Similarity.

R. K. Mishra et al. [7] presents an alternative inter-passage dependent clustering strategy that can cluster portions of documents based on the similarity. The data will be a set of documents containing multi-topic segments gathered from the internet. The section score of the sections of the reports is computed by SentiWordNet. On the intra-document level, section-based clustering will be implemented based on the section score.

N. Abd Rahman et al. [4] concentrates on document clustering for the compilation of Malay tests It contains 2028 Hadith documents from the journal Sahih Bukhari that have been adapted into Malay. The results of using the Complete Linkage Clustering algorithm with the Cosine Coefficient on Malay translated Hadith documents are presented in their paper.

N. Shah et al. [5] showed The theory of linguistics is expected to be mainly based on distributed document clustering using Hadoop and MapReduce. The distributed version of two well-known algorithms, K-Means and Bisecting K-Means, is executed; testing and contrasting both of these algorithms on single-node and multi-node Hadoop clusters for comprehensiveness and consistency is done.

Anjali Gunta et al. [1] introduces two linked document clustering strategies and document clustering based on Multiview point-based similarity nature. The current clustering dissimilarity or similarity live document only uses one viewpoint: the source, indicating how it only uses one point of comparison. They, on the other hand, employ a range of references.

CHAPTER 3

METHODOLOGY

3.1. AIM OF THE PROJECT

The aim of the project is 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.

3.2 SYSTEM REQUIREMENTS

3.2.1 SOFTWARE REQUIREMENTS

- Operating System : Windows 10
- Python :3.8
- Packages : Numpy, Pandas, Matplotlib, Scipy etc.

3.2.2 HARDWARE REQUIREMENTS:

- Hard disk :512GB
- Processor : i7
- Ram :16GB

3.3 OVERVIEW OF THE PLATFORM

3.3.1 Python

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code. It is a programming language that lets you work quickly and integrate systems more efficiently.

Python is used for:

- web development (server-side),

- software development,
- mathematics,
- System

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- Python 2.0 was released in 2000, and the 2.x versions were the prevalent releases until December 2008. At that time, the development team made the decision to release version 3.0, which contained a few relatively small but significant changes that were not backward compatible with the 2.x versions.

Python 2 and 3 are very similar, and some features of Python 3 have been backported to Python 2. But in general, they remain not quite compatible.

- Both Python 2 and 3 have continued to be maintained and developed, with periodic release updates for both. As of this writing, the most recent versions available are 2.7.15 and 3.6.5. However, an official End of Life date of 9 January 1, 2020 has been established for Python 2, after which time it will no longer be maintain
- Python is still maintained by a core development team at the Institute, and Guido is still in charge, having been given the title of BDFL (Benevolent Dictator For Life) by the Python community. The name Python, by the way, derives not from the snake, but from the British comedy troupe Monty Python's Flying Circus, of which Guido was, and presumably still is, a fan. It is common to find references to Monty Python sketches and movies scattered throughout the Python documentation.
- It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- Python was designed to for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Python is Interpreted

- Many languages are compiled, meaning the source code you create needs to be translated into machine code, the language of your computer's

processor, before it can be run. Programs written in an interpreted language are passed straight to an interpreter that runs them directly.

- This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.
- One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.
- In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The expediency of coding in an interpreted language is typically worth it for most applications.
- For all its syntactical simplicity, Python supports most constructs that would be expected in a very high-level language, including complex dynamic data types, structured and functional programming, and object-oriented programming.
- Additionally, a very extensive library of classes and functions is available that provides capability well beyond what is built into the language, such as database manipulation or GUI programming.
- Python accomplishes what many programming languages don't: the language itself is simply designed, but it is very versatile in terms of what you can accomplish with it.

3.3.2 Machine Learning using Python

Clustering in Machine Learning

It is basically a type of unsupervised learning method. An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labelled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples. Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

For ex– The data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.

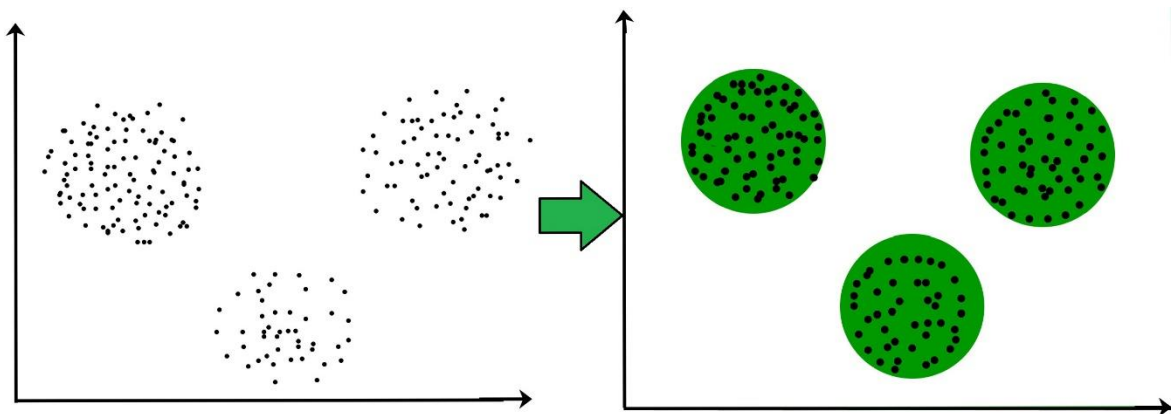


Fig 3.1: Distinguishing of clusters

It is not necessary for clusters to be a spherical. Such as :

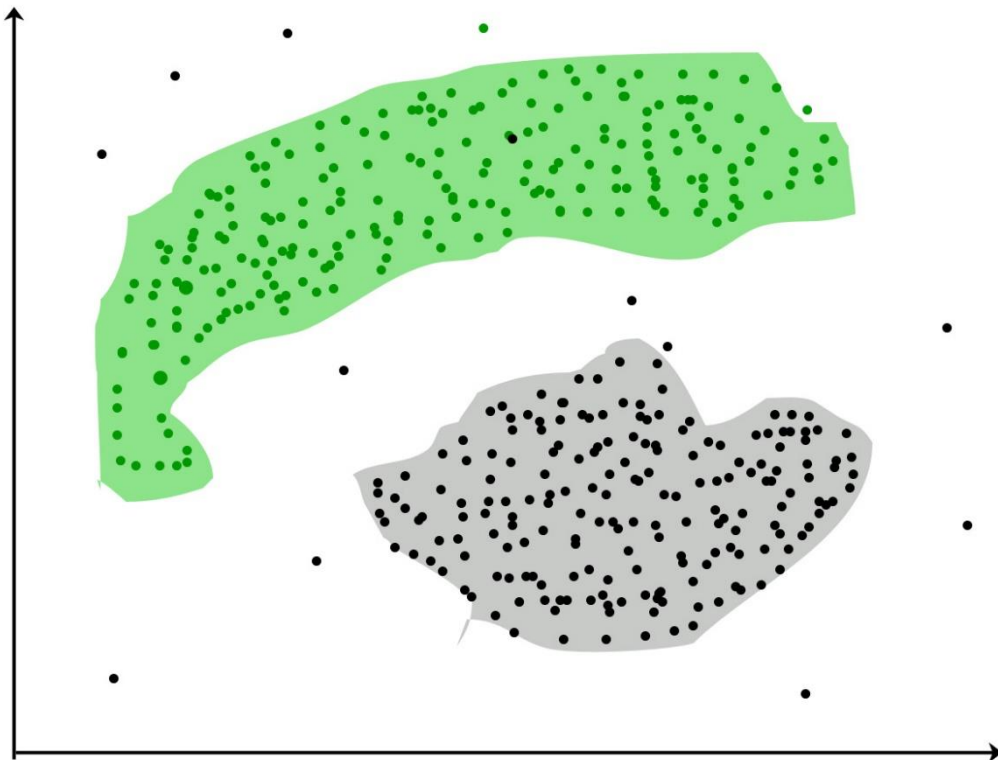


Fig 3.2 :Density-based Spatial Clustering

DBSCAN: Density-based Spatial Clustering of Applications with Noise

These data points are clustered by using the basic concept that the data point lies within the given constraint from the cluster centre. Various distance methods and techniques are used for calculation of the outliers.

Clustering is very much important as it determines the intrinsic grouping among the unlabelled data present. There are no criteria for a good clustering. It depends on the user, what is the criteria they may use which satisfy their need. For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding “natural clusters” and describe their unknown properties (“natural” data types), in finding useful and suitable groupings (“useful” data classes) or in finding unusual data objects (outlier detection). This algorithm must make some assumptions which constitute the similarity of points and each assumption make different and equally valid clusters.

The Stages of the Process of Clustering:

It is important to emphasize that getting from a collection of documents to a clustering of the collection, is not merely a single operation, but is more a process in multiple stages. These stages include more traditional information retrieval operations such as crawling, indexing, weighting, filtering etc. Some of these other processes are central to the quality and performance of most clustering algorithms, and it is thus necessary to consider these stages together with a given clustering algorithm to harness its true potential. We will give a brief overview of the clustering process,

the offline clustering process into the four stages outlined below:

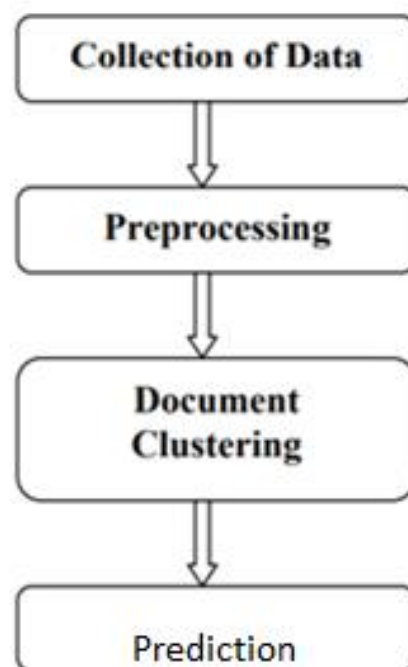


Fig 3.3: Clustering process

Clustering Algorithms:

K-means clustering algorithm – It is the simplest unsupervised learning algorithm that solves clustering problem- kmeans algorithm partition n observations into k clusters where each observation belongs to the cluster with the nearest mean

serving as a prototype of the cluster.

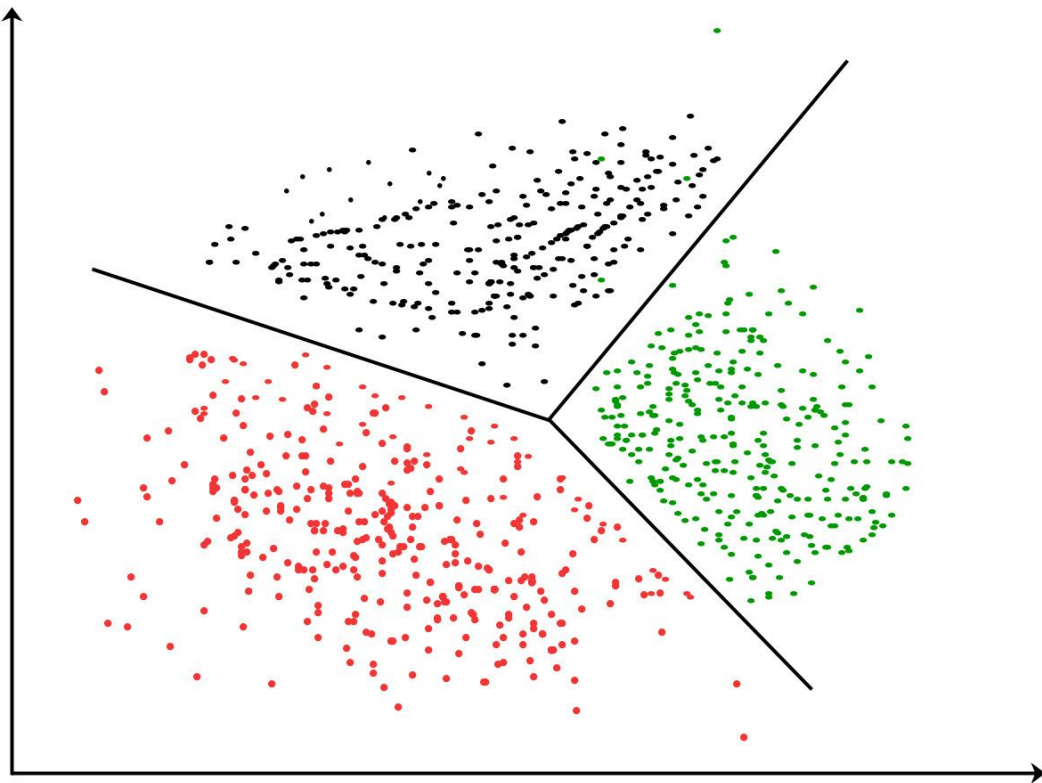


Fig 3.4: K-means algorithm partition n observations into k clusters

Applications of Clustering in different fields

- **Marketing:** It can be used to characterize & discover customer segments for marketing purposes.
- **Biology:** It can be used for classification among different species of plants and animals.
- **Libraries:** It is used in clustering different books on the basis of topics and information.
- **Insurance:** It is used to acknowledge the customers, their policies and identifying the frauds.
- **City Planning:** It is used to make groups of houses and to study their values based on their geographical locations and other factors present.
- **Earthquake studies:** By learning the earthquake-affected areas we can determine the dangerous zones.

CHAPTER 4

MODULE DESCRIPTION

4.1 SYSTEM ARCHITECTURE:

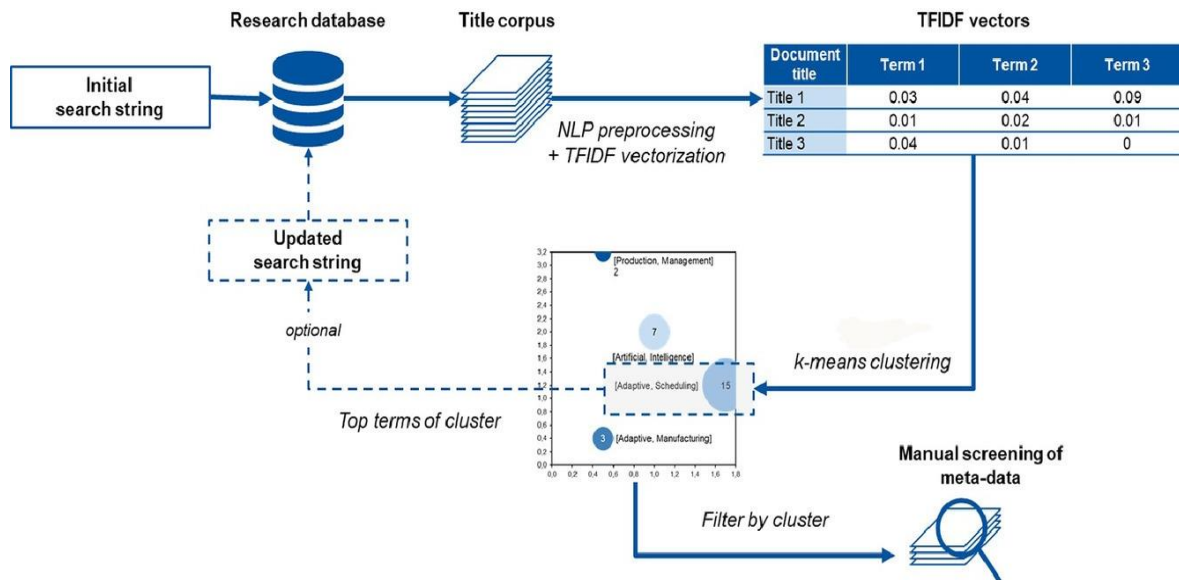


FIG 4.1 System Architecture

4.2 MODULE DESCRIPTION

COLLECTION OF DATA

In this project, we chose to focus on The White House and a group of leading research organisations have developed the COVID-19 Open Research Dataset in response to the COVID-19 pandemic (CORD-19). CORD-19 is a database of over 51,000 scientific papers about COVID-19, SARS-CoV-2, and associated coronaviruses, including over 40,000 with full text. This publicly accessible dataset is made available to the entire world. This freely available dataset is provided to the global research community to apply recent advances in natural language processing and other AI techniques to generate new insights in support of the ongoing fight against this infectious disease. There is a growing urgency for these approaches because of the rapid acceleration in new coronavirus literature, making it difficult for the medical research community to keep up.

LOADING

Firstly, we start by uploading the csv file, which is the metadata. Then we upload and read the zip file that contain all the JSON file which has the full text information of the articles. With the help of helper function, the JSON files can be read.

PREPROCESSING

Pre-processing simply refers to perform series of operations to transform or change data. It is transformation applied to our data before feeding it to algorithm. Data processing refers to perform operations on data to retrieve, transform, or change data, especially by computer. It is technique that is used to convert raw data into clean data set. In other words, whenever data is gathered from different sources, it is collected in raw format, which is not feasible for analysis. Then it converts raw format into readable format (graphs, documents, etc.), so that it can be interpreted by computers and utilized by employees throughout an organization. Data goes through a series of steps during pre-processing:

Data Cleaning: Data is cleansed through processes such as filling in missing values or deleting rows with missing data, smoothing the noisy data, or resolving the inconsistencies in the data.

Smoothing noisy data is particularly important for ML datasets, since machines cannot make use of data they cannot interpret. Data can be cleaned by dividing it into equal size segments that are thus smoothed (binning), by fitting it to a linear or multiple regression function (regression), or by grouping it into clusters of similar data (clustering).

Data inconsistencies can occur due to human errors (the information was stored in a wrong field). Duplicated values should be removed through deduplication to avoid giving that data object an advantage (bias).

Data Integration: Data with different representations are put together and conflicts within the data are resolved.

Data Transformation: Data is normalized and generalized. Normalization is a process that ensures that no data is redundant, it is all stored in a single place, and all the dependencies are logical.

Data Reduction: When the volume of data is huge, databases can become slower, costly to access, and challenging to properly store. Data reduction step aims to present a reduced representation of the data in a data warehouse.

There are various methods to reduce data. For example, once a subset of relevant attributes is chosen for its significance, anything below a given level is discarded. Encoding mechanisms can be used to reduce the size of data as well. If all original data can be recovered after compression, the operation is labelled as lossless.

If some data is lost, then it's called a lossy reduction. Aggregation can also be used, for example, to condense countless transactions into a single weekly or monthly value, significantly reducing the number of data objects.

Data Discretization: Data could also be discretized to replace raw values with interval levels. This step involves the reduction of a number of values of a continuous attribute by dividing the range of attribute intervals.

Data Sampling: Sometimes, due to time, storage or memory constraints, a dataset is too big or too complex to be worked with. Sampling techniques can be used to select and work with just a subset of the dataset, provided that it has approximately the same properties of the original one.

With pre-processing the data, we need to clean-up the text to improve any clustering or classification efforts. Then add new columns for word counts for abstract, for body text and for count of unique words used in body text.

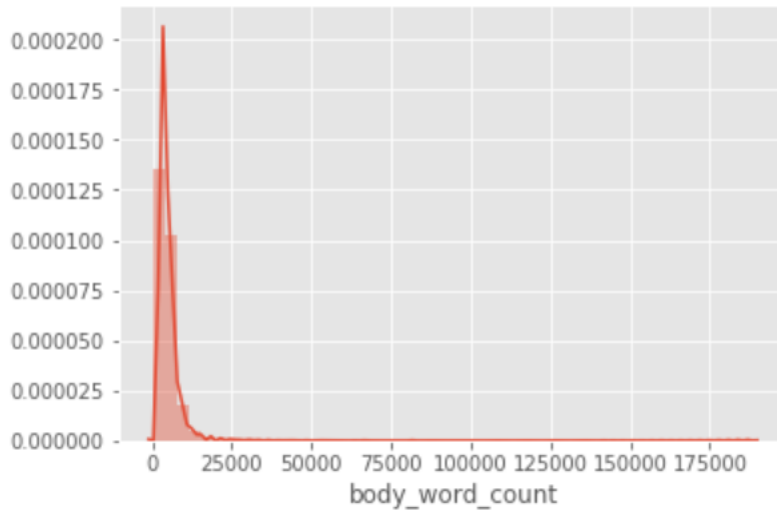


Fig 4.2: Word count of body text

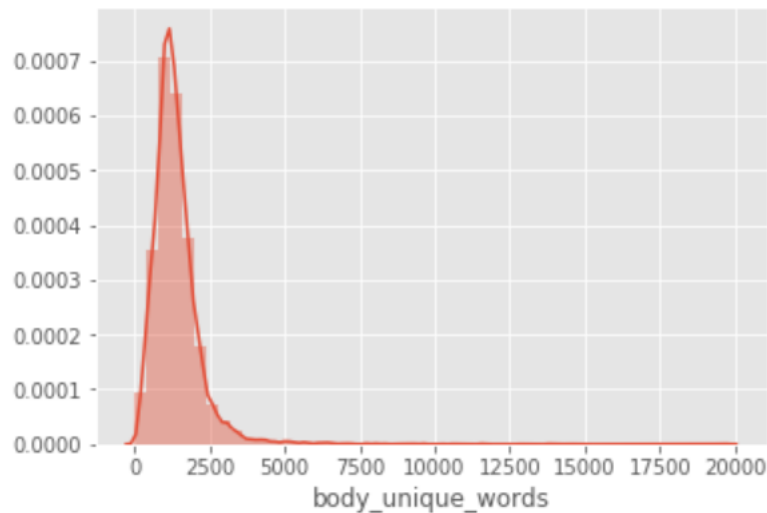


Fig 4.3: Count of unique words in the body text

Set() is used to convert the iterations of words to distinct or unique value. Part of pre-processing also includes handling duplicates which may have been caused because of author submitting the article to multiple journals. Using Describe() function, we come to know that majority of the body text or abstract are unique yet there are no duplicate.

Then we will be dropping the null values like articles with missing abstract and body text. We take a sample of 10k papers, from those papers, we drop 699 article that don't provide abstract or body text. Next we are going to determine the language of

each paper in the data frame. Then we drop articles that are not in English, since translating those articles to English will lose its real semantic meaning

```
Total: 9301
```

```
{'ca': 1, 'de': 22, 'en': 9087, 'es': 80, 'fr': 97, 'it': 3, 'nl': 9, 'pt': 2}
```

FIG language distribution in the dataset

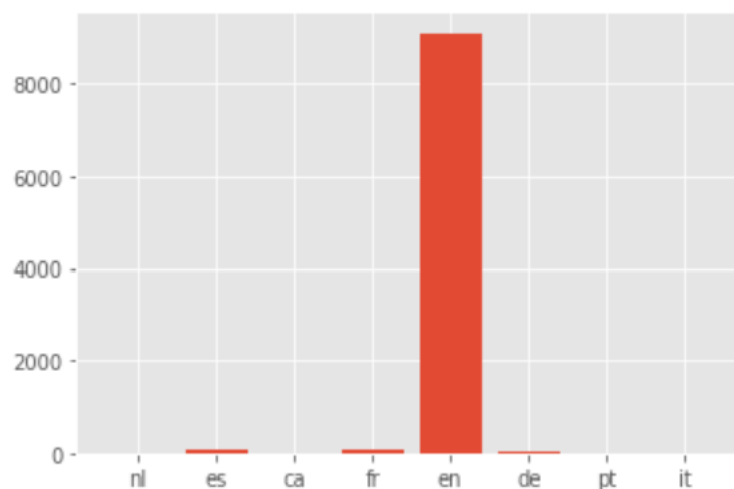


Fig 4.4 distribution of languages in the dataset

Then we remove the stop words that are not important but are constantly repeated. The stopwords are used in everyday English text. Research papers will often frequently use words that don't actually contribute to the meaning and are not considered everyday stopwords.

Next we need to process the text data. For this purpose we will be using the spacy library. This function will convert text to lower case, remove punctuation, and find and remove stopwords. For the parser, we will use `en_core_sci_lg`. This is a model for processing biomedical, scientific or clinical text. Then we convert texts to lowercase and remove the punctuation.

VECTORIZATION

The next step is to vectorize the text 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. From all the vectorizers, we will be use TF-IDF, also know as Term Frequency Inverse Document Frequency. This vectorizer parses text and returns a float value as a vector that indicates the frequency of a word in that document or all of documents. To create Vector space model, a subset of the tokens from the dataset is required which is known as Feature vector. We have used frequency-based method for feature vector extraction. c) Vector space Model and Similarity Measure: It is also known as TF-IDF model i.e. term frequency inverse document frequency model. It is the standard retrieval technique used in text mining area. In this model, each document is represented as an n-dimensional vector using the feature vector. The value of each element in the vector reflects the importance of the corresponding feature in the document. Text documents are converted into machine acceptable, mathematical representations after the transformation. The similarity between documents can be measured by calculating the distance between document vectors [1]. Now that D is a vector => Given a doc, find vectors (docs) “near” it. Intuition: Documents that are “close together” in vector space talk about the same thing .Since an article has from 5000 to 20,000 words, it’s a lot to parse all of the words. So we will provide top 4096 (2 ** 12). The formula for TF is:

TF (t) = (Number of times term t appears in the document)/(Total number of terms in the document)

The formula for Inverse Document frequency is:

IDF (t) = log (Total amount of documents/Number of documents with the term t in it)

Both TF and IDF multiplied to give a vector and determines the frequency and importance of the term t.

DIMENSIONALITY REDUCTION

Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. These new transformed features

are called the **Principal Components**. It is one of the popular tools that is used for exploratory data analysis and predictive modelling. It is a technique to draw strong patterns from the given dataset by reducing the variances. PCA generally tries to find the lower-dimensional surface to project the high-dimensional data. In other words. It reduces the multi-dimensional to two-dimensional.

PCA works by considering the variance of each attribute because the high attribute shows the good split between the classes, and hence it reduces the dimensionality. Some real-world applications of PCA are image processing, movie recommendation system, optimizing the power allocation in various communication channels. It is a feature extraction technique, so it contains the important variables and drops the least important variable.

There are certain steps we have followed for dimensionality reduction using PCA:

Getting the dataset

Firstly, we need to take the input dataset and divide it into two subparts X and Y, where X is the training set, and Y is the validation set.

Representing data into a structure

Now we will represent our dataset into a structure. Such as we will represent the two-dimensional matrix of independent variable X. Here each row corresponds to the data items, and the column corresponds to the Features. The number of columns is the dimensions of the dataset.

Standardizing the data

In this step, we will standardize our dataset. Such as in a particular column, the features with high variance are more important compared to the features with lower variance.

If the importance of features is independent of the variance of the feature, then we will divide each data item in a column with the standard deviation of the column. Here we will name the matrix as Z.

Calculating the Covariance of Z

To calculate the covariance of Z, we will take the matrix Z, and will transpose it.

After transpose, we will multiply it by Z. The output matrix will be the Covariance matrix of Z.

Calculating the Eigen Values and Eigen Vectors

Now we need to calculate the eigenvalues and eigenvectors for the resultant covariance matrix Z. Eigenvectors or the covariance matrix are the directions of the axes with high information. And the coefficients of these eigenvectors are defined as the eigenvalues.

Sorting the Eigen Vectors

In this step, we will take all the eigenvalues and will sort them in decreasing order, which means from largest to smallest. And simultaneously sort the eigenvectors accordingly in matrix P of eigenvalues. The resultant matrix will be named as P*.

Calculating the new features Or Principal Components

Here we will calculate the new features. To do this, we will multiply the P* matrix to the Z. In the resultant matrix Z*, each observation is the linear combination of original features. Each column of the Z* matrix is independent of each other.

Remove less or unimportant features from the new dataset.

The new feature set has occurred, so we will decide here what to keep and what to remove. It means, we will only keep the relevant or important features in the new dataset, and unimportant features will be removed out.

K MEANS CLUSTERING ALGORITHM

Through PCA it keeps majority of the data while removing the outliers without losing its semantic meaning. The reduced data is used by k-means. The k-means clustering algorithm is known to be efficient in clustering large data sets. This algorithm is one of the simplest and the best-known unsupervised learning algorithms.

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering. The algorithm takes the unlabelled dataset as input, divides the dataset into k-number of clusters, and repeats the

process until it does not find the best clusters. The value of k should be predetermined in this algorithm. It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters. The k -means clustering algorithm mainly performs two tasks:

- Determines the best value for K centre points or centroids by an iterative process.
- Assigns each data point to its closest k -centre. Those data points which are near to the particular k -centre, create a cluster.

K -Means Clustering groups the unlabelled dataset into different clusters. K defines the number of pre-defined clusters that need to be created in the process, as if $K=2$, there will be two clusters, and for $K=3$, there will be three clusters, and so on.

The K -Means algorithm aims to partition a set of objects, based on their attributes/features, into k clusters, where k is a predefined constant. initialize any random points called as the centroids of the cluster. The algorithm defines k centroids, one for each cluster. While initializing you must take care that the centroids of the cluster must be less than the number of training data points. This algorithm is an iterative algorithm hence the next two steps are performed iteratively. The centroid of a cluster is formed in such a way that it is closely related, in terms of similarity (where similarity can be measured by using different methods such as Euclidean distance or Extended Jaccard) to all objects in that cluster [9]. Technically, what k -means is interested in, is the variance. It minimizes the overall variance, by assigning each object to the cluster such that the variance is minimized. Coincidentally, the sum of squared deviations, one objects contribution to the total variance, over all dimensions is exactly the definition of squared Euclidean distance.

K -means is applied where a datapoint is made as the centroid and mean distance is taken and clustered. The centroids are incrementally modified. As the clusters formed in the above step are not optimized so we need to form optimized clusters. For this, we need to move the centroids iteratively to a new location. Take data

points of one cluster, compute their average and then move the centroid of that cluster to this new location. Repeat the same step for all other clusters. The above two steps are done iteratively until the centroids stop moving i.e. they do not change their positions anymore and have become static. Once this is done the k-means algorithm is termed to be converged. Now this algorithm has converged and distinct clusters are formed and clearly visible. This algorithm can give different results depending on how the clusters were initialized in the first step.

But before applying k-means, we need to find the optimal k value to determine right number of clusters give the most accurate results. For that, we use the elbow method. Distortion is calculated and plotted against k-values. An elbow is formed, that represents around which number of clusters gives accurate results. For simplicity we will be taking 20 as our k value. Then we train k-mean algorithm and predict labels for our cluster and save them in the object 'y_preds'.

In the project 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.

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.

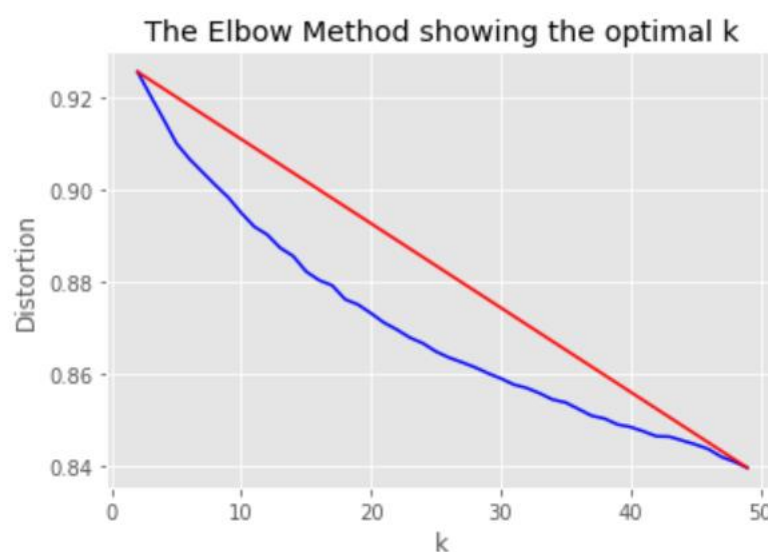


Fig 4.5 Elbow method

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 $k=20$

CLUSTERING AND CLASSIFYING LABELS

A plot is presents but it is bland. The distinguish between them, using predicted labels by k -means, we set palette and hue as y -preds. Clusters that are obtained as a result of the clustering process need to be labelled appropriately in order to understand the purpose of each cluster and to evaluate the effectiveness of clustering. Cluster labelling is based on selecting words from each cluster and then use them to label the clusters. There are three ways of selecting words for cluster labelling. The first method (discriminative labelling) assumes the existence of a document hierarchy, either manually constructed and/or populated, or a hierarchy resulting from application of a hierarchical clustering algorithm.

Using chi square tests of independence at each node in the hierarchy starting from the root, we determine a set of words that are equally likely to occur in any of the children of a current node. Such words are general for all of the sub-trees of a current node, and are excluded form the nodes below. The second method (non-discriminative labelling) selects words which both occur frequently in a cluster or effectively discriminate the given cluster from the other clusters. While the third method, using titles for labelling clusters, is based on the titles 8 of the documents within each cluster to find the most representative words for the cluster. The work in [25] briefly describe a technique to label clusters based on how many times a feature is used in a cluster. By utilizing this information, and also drawing on knowledge of the code, short titles are manually selected for the obtained clusters. Although labelling is performed manually, they point out that the automatically developed feature summary of each cluster makes the labelling process much easier. In 2001 Tzerpos et al. [26] emphasizes that a clustering algorithm should have certain features to make its output easier to comprehend. These features include bounded cluster cardinality, which ensures that any single cluster does not contain a very large number of entities, and effective cluster naming. They use a pattern-based

approach to recognizing familiar subsystem structures within large systems. The identified patterns are expected to occur in large systems with around 100 source files. The same pattern-based approach is used for cluster labelling. In 2003 Tonella et al. [27] describe the use of keywords within web pages to cluster and label similar pages. Both single words and a contiguous sequence of two words i.e., bigrams are considered as representative keywords of a webpage. Clustering as well as cluster labelling are carried out on the basis of keywords within a webpage. Cluster labels are ranked according to inverse keyword frequency. We will be using t-SNE also known as t-Distributed Stochastic Neighbour Embedding is not only used for dimensionality reduction but also does a great job at clustering and visualization of the plot. Data before PCA is used and reduced and plotted. t-SNE places similar articles together in a cluster and dissimilar articles further away from others.

A. t-Distributed Stochastic Neighbor Embedding (t-SNE)

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a non-linear technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets. It is extensively applied in image processing, NLP, genomic data and speech processing. To keep things simple, here's a brief overview of working of t-SNE:

- The algorithms start by calculating the probability of similarity of points in high-dimensional space and calculating the probability of similarity of points in the corresponding low-dimensional space. The similarity of points is calculated as the conditional probability that a point A would choose point B as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian (normal distribution) centered at A.
- It then tries to minimize the difference between these conditional probabilities (or similarities) in higher-dimensional and lower-dimensional space for a perfect representation of data points in lower-dimensional space.
- To measure the minimization of the sum of difference of conditional probability t-SNE minimizes the sum of Kullback-Leibler divergence of overall data points using a gradient descent method.

In simpler terms, t-Distributed stochastic neighbor embedding (t-SNE) minimizes the divergence between two distributions: a distribution that measures pairwise similarities of the input objects and a distribution that measures pairwise similarities of the corresponding low-dimensional points in the embedding.

In this way, t-SNE maps the multi-dimensional data to a lower dimensional space and attempts to find patterns in the data by identifying observed clusters based on similarity of data points with multiple features. However, after this process, the input features are no longer identifiable, and you cannot make any inference based only on the output of t-SNE. Hence it is mainly a data exploration and visualization technique.

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

So that step took a while! Let's take a look at what our data looks like when compressed to 2 dimensions.

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.

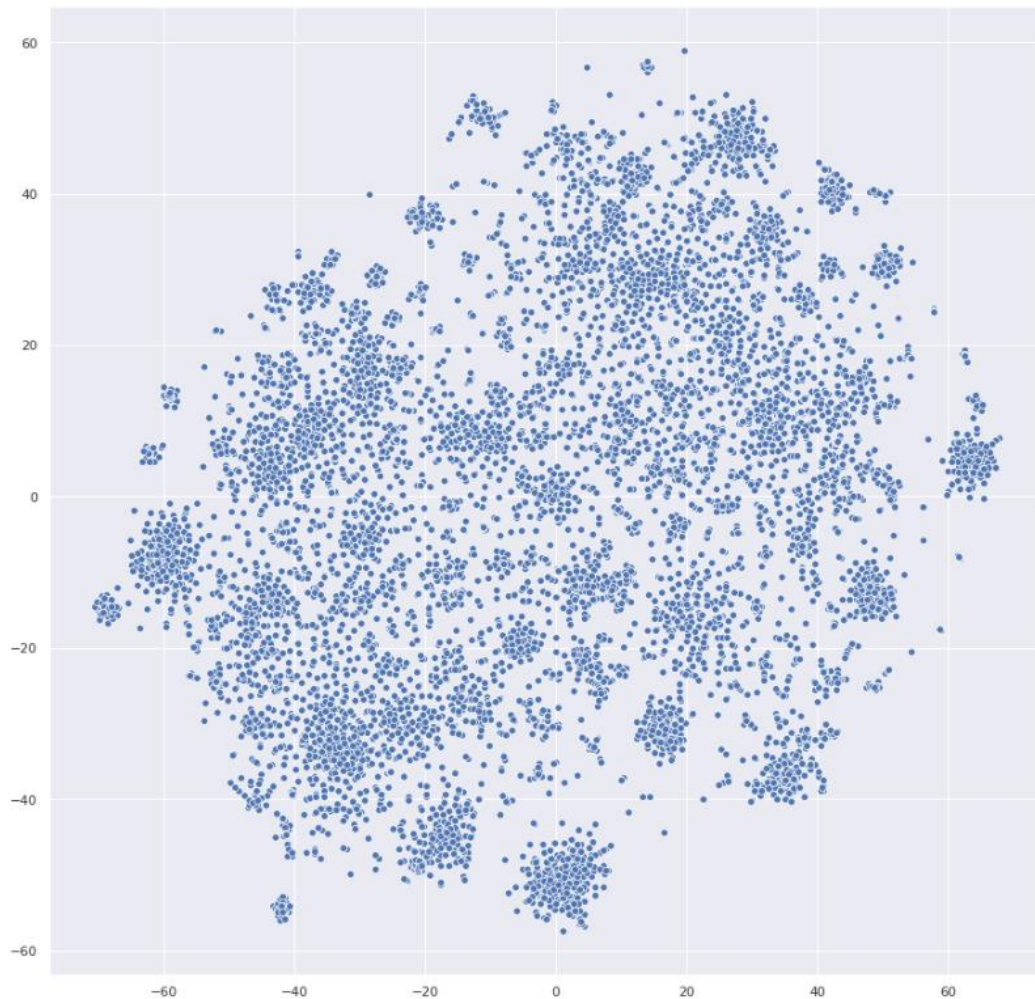


Fig 4.6 t-SNE without labels

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 independetly. 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.

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.

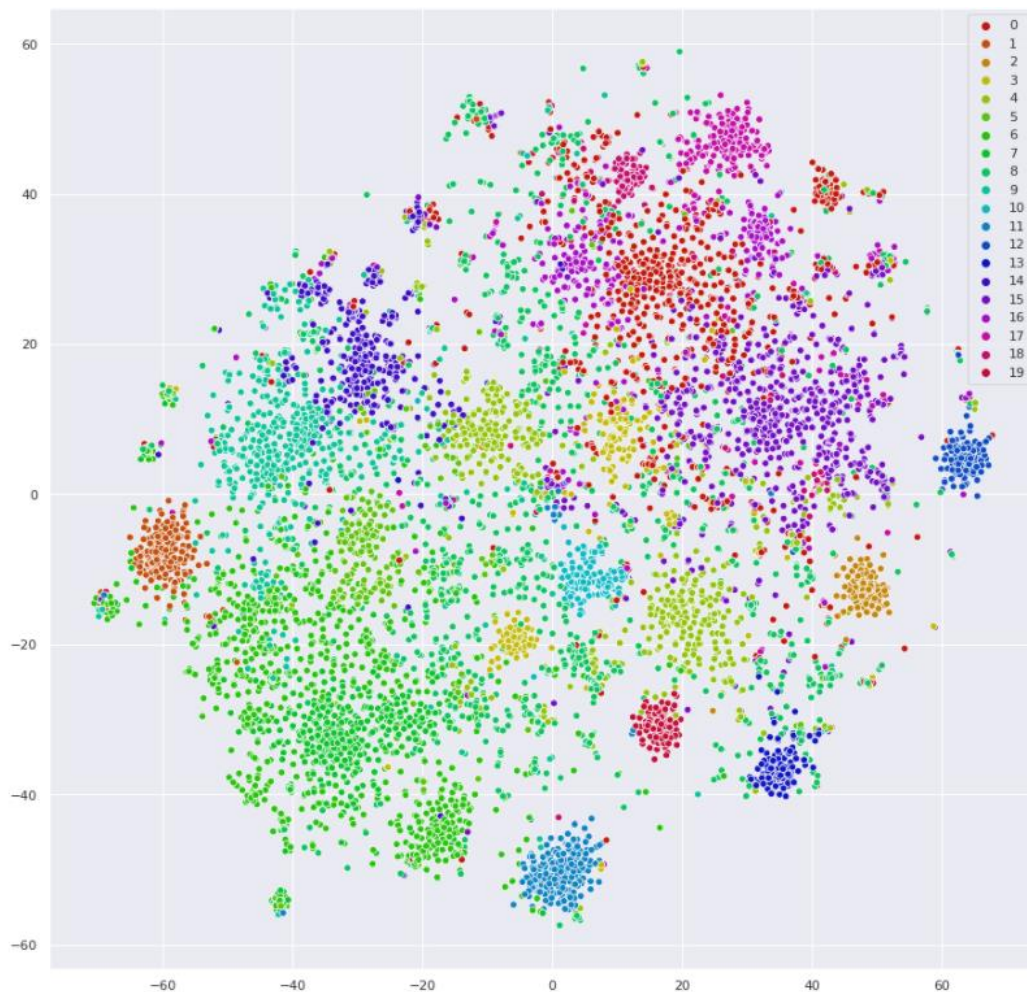


fig 4.7 T-sne with k means labels

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 unnaparent to humans. This may highlight hidden shared information and advance further research.

TOPIC MODELLING

Topic modelling is something we do to put on a label on a cluster. For topic modelling, we will be using Latent Dirichlet Allocation (LDA). It is a model that assigns a topic based on a group of documents, with a bunch of related keywords. For each iteration or cluster, we will vectorize them and LDA model will be applied and it extracts keywords. We will then plot the clusters using bokeh plot. the slider is used to pick a certain cluster and search bar to search for a particular keyword.

Now we will attempt to find the most significant words in each clusters. 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.

Topic modeling is an unsupervised machine learning technique that's capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize a set of documents.

It's simple, really. Topic modeling involves counting words and grouping similar word patterns to infer topics within unstructured data. Let's say you're a software company and you want to know what customers are saying about particular features of your product. Instead of spending hours going through heaps of feedback, in an attempt to deduce which texts are talking about your topics of interest, you could analyze them with a topic modeling algorithm.

By detecting patterns such as word frequency and distance between words, a topic model clusters feedback that is similar, and words and expressions that appear most often. With this information, you can quickly deduce what each set of texts are talking about. Remember, this approach is 'unsupervised' meaning that no training is required.

Now, let's say you train a model to detect specific topics. That's a whole different kettle of fish, and a step that's needed for topic classification algorithms – a supervised technique. Let's compare the two topic analysis algorithms to further understand the differences between them.

For topic modeling, we will use LDA (Latent Dirichlet Allocation).

LDA stands for Latent Dirichlet Allocation. As time is passing by, data is increasing exponentially. Most of the data is unstructured and a few of them are unlabeled. It is a tedious task to label each and every data manually. How can we label such a huge amount of data if not manually? Here comes the LDA to our rescue. LDA is one of the topic modeling techniques which is used to analyze a huge amount of data, cluster them into similar groups, and label each group. It should be noted that LDA technique is for unsupervised learning which is used to label the data by grouping them into similar topics. Unlike K-Means clustering and other clustering techniques which uses the concept of distance between cluster center, LDA works on the probability distribution of topics belonging to the document.

1) Latent Dirichlet Allocation

To understand how topic modeling works, we'll look at an approach called Latent Dirichlet Allocation (LDA).

This is a popular approach that is widely used for topic modeling across a variety of applications. It has good implementations in coding languages such as Java and Python and is therefore easy to deploy. LDA was developed in 2003 by researchers David Blei, Andrew Ng and Michael Jordan. Its simplicity, intuitive appeal and effectiveness have led to strong support for its use. LDA topic modeling discovers topics that are hidden (latent) in a set of text documents. It does this by inferring possible topics based on the words in the documents. It uses a generative probabilistic model and Dirichlet distributions to achieve this.

The inference in LDA is based on a Bayesian framework. This allows the model to infer topics based on observed data (words) through the use of conditional probabilities. A generative probabilistic model works by observing data, then generating data that's similar to it in order to understand the observed data. This is a powerful way to analyze data and goes beyond mere description—by learning how to generate observed data, a generative model learns the essential features that characterize the data.

In LDA, the generative process is defined by a joint distribution of hidden and observed variables. A Dirichlet distribution can be thought of as a *distribution over distributions*.

In the case of LDA, if we have K topics that describe a set of documents, then the mix of topics in each document can be represented by a K -nomial distribution, a form of multinomial distribution.

A multinomial distribution is a generalization of the more familiar binomial distribution (which has 2 possible outcomes, such as in tossing a coin). A K -nomial distribution has K possible outcomes (such as in a K -sided dice).

In LDA, the Dirichlet is a probability distribution over the K -nomial distributions of topic mixes. There is another Dirichlet distribution used in LDA—a Dirichlet over the words in each topic. So, LDA uses two Dirichlet distributions in its algorithm.

By using a generative process and Dirichlet distributions, LDA can better generalize to new documents after it's been trained on a given set of documents. This is an improvement on predecessor models (such as pLSI).

2) LDA preliminaries

The first thing to note with LDA is that we need to decide the number of topics, K , in advance.

By choosing K , we are saying that we believe the set of documents we're analyzing can be described by K topics.

If we're not quite sure what K should be, we can use a trial-and-error approach, but clearly the need to set K is an important assumption in LDA.

When analyzing a set of documents, the total set of words contained in all of the documents is referred to as the *vocabulary*.

Besides K , there are other parameters that we can set in advance when using LDA, but we often don't need to do so in practice—popular implementations of LDA assume default values for these parameters if we don't specify them.

We'll look at some of these other parameters later.

The second thing to note with LDA is that once the K topics have been identified, LDA does not tell us anything about the topics other than showing us the distribution of words contained within them (ie. the probability of each word in the vocabulary appearing in the topic).

We therefore need to use our own interpretation of the topics in order to understand what each topic is about and to give each topic a name.

The above two characteristics of LDA suggest that some domain knowledge can be helpful in LDA topic modeling.

Although it's not required for LDA to work, domain knowledge can help us choose a sensible number of topics (K) and interpret the topics in a way that's useful for the analysis being done.

3) The LDA algorithm—Step by step

Let's now look at the algorithm that makes LDA work—it's basically an iterative process of topic assignments for each word in each document being analyzed.

Recall that LDA identifies the latent topics in a set of documents.

What this means is that for each document, LDA will generate the topic mix, or the distribution of topics for each document.

All documents share the same K topics, but with different proportions (mixes).

Having chosen a value for K, the LDA algorithm works through an iterative process as follows:

Step 1

Initialize the model:

- Randomly assign a topic to *each word in each document*
- Note that after this random assignment, two frequencies can be computed—

1. the counts (frequency distribution) of topics in each document, call this *topic frequency*
2. the counts (frequency distribution) of words in each topic, call this *word frequency*

Step 2

Update the topic assignment for a *single word in a single document*:

- Choose a word in a document
 - Un-assign its assigned topic (ie. un-assign the topic that was randomly assigned during the initialization step)
 - Re-assign a topic to the word, given (ie. conditional upon) all other topic assignments for all other words in all documents, by considering—
 1. the popularity of each topic in the document, ie. how many times the document uses each topic, measured by the frequency counts calculated during initialization (topic frequency) *and* a Dirichlet-generated multinomial distribution over topics for each document, and
 2. the popularity of the word in each topic, ie. how many times each topic uses the word, measured by the frequency counts calculated during initialization (word frequency), *and* a Dirichlet-generated multinomial distribution over words for each topic
 - Multiply 1. and 2. to get the conditional probability that the word takes on each topic
 - Re-assign the word to the topic with the largest conditional probability

Step 3

Repeat Step 2 for all words in all documents.

Step 4

Iterate.

How LDA topic modeling works: A 4-step iterative algorithm using Dirichlet distributions

In Step 2 of the algorithm, you'll notice the use of two Dirichlets—what role do they serve?

As mentioned, by including Dirichlets in the model it can better generalize to new documents.

To understand why Dirichlets help with better generalization, consider the case where the frequency count for a given topic in a document is zero, eg. if the topic does not appear in a given document after the random initialization.

But the topic may actually have relevance for the document.

By including a Dirichlet, which is a probability distribution over the K-nomial topic distribution, a non-zero probability for the topic is generated.

Hence, the topic may be included in subsequent updates of topic assignments for the word (Step 2 of the algorithm).

This additional variability is important in giving all topics a chance of being considered in the generative process, which can lead to better representation of new (unseen) documents.

4) Why does the LDA algorithm work?

Step 2 of the LDA algorithm calculates a conditional probability in two components—one relating to the distribution of topics in a document and the other relating to the distribution of words in a topic.

Hence, each word's topic assignment depends on *both* the probability of the topic in the document and the probability of the word in the topic.

In this way, words will move together within a topic based on the *suitability* of the word for the topic and also the *suitability* of the topic for the document (which considers all other topic assignments for all other words in all documents).

Note that *suitability* in this sense is determined solely by frequency counts and Dirichlet distributions and not by semantic information. In this way, the observed structure of the document informs the discovery of latent relationships, and hence the discovery of latent topic structure.

In LDA, each document can be described by a distribution of topics and each topic can be described by a distribution of words. First we will create 20 vectorizers, one for each of our cluster labels. Now we will vectorize the data from each of our clusters

Topic modeling will be performed through the use of Latent Dirichlet Allocation (LDA). This is a generative statistical model that allows sets of words to be explained by a shared topic

For each cluster, we had created a corresponding LDA model in the previous step. We will now `fit_transform` all the LDA models on their respective cluster vectors

Extracts the keywords from each cluster

5) ***Save current outputs to file***

Re-running some parts of the notebook (especially vectorization and t-SNE) are time intensive tasks. We want to make sure that the important outputs for generating the bokeh plot are saved for future use.

CHAPTER 5

RESULT AND DISCUSSION

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 a 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 modelling 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 colours) 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.

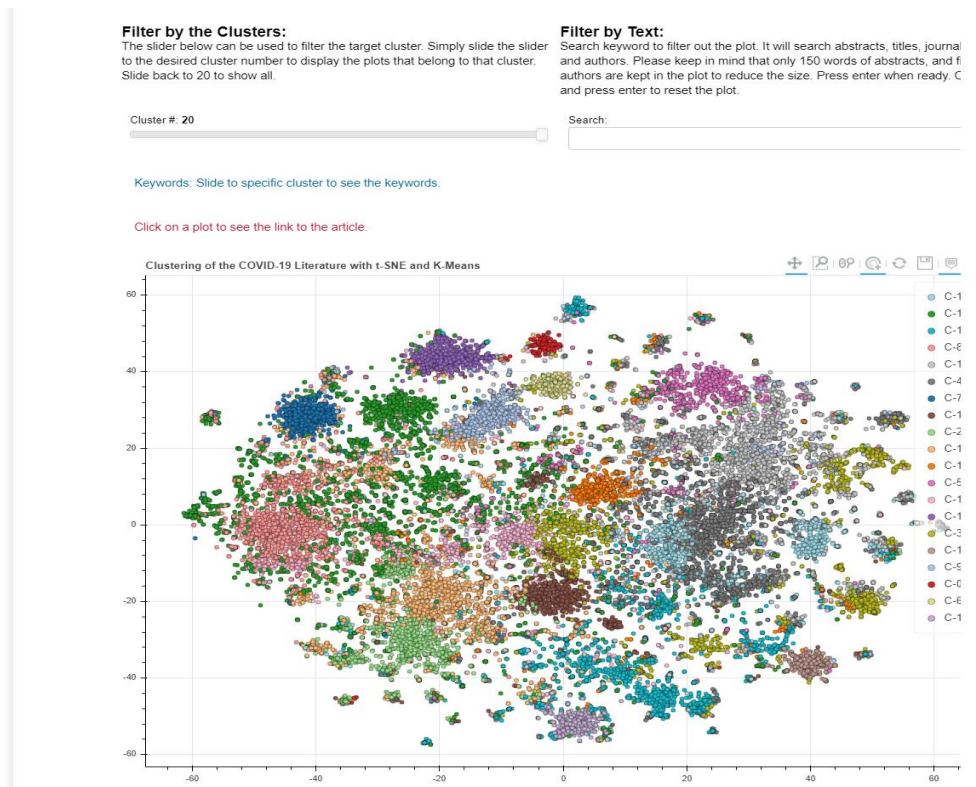


Fig 5.1 bokeh plot

In the search tool, when you search for a keyword, datapoints or clusters that belong to that topic or keyword will appear. Here, we have searched for the term “vaccine”.

Filter by the Clusters:

The slider below can be used to filter the target cluster. Simply slide the slider to the desired cluster number to display the plots that belong to that cluster. Slide back to 20 to show all.

Cluster #: 20

**Filter by Text:**

Search keyword to filter out the plot. It will search abstracts, titles, journal and authors. Please keep in mind that only 150 words of abstracts, and 10 authors are kept in the plot to reduce the size. Press enter when ready. C and press enter to reset the plot.

Search:

vaccine

Keywords: Slide to specific cluster to see the keywords.

Click on a plot to see the link to the article.

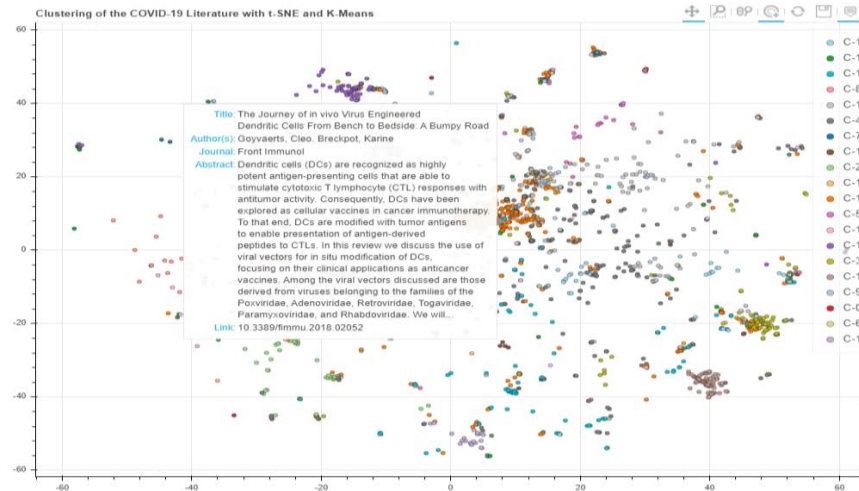


Fig 5.3 Description

In this below figure, when you bring the cursor and hover over the datapoints, you get to see the descriptive information about a singular article.

Filter by the Clusters:

The slider below can be used to filter the target cluster. Simply slide the slider to the desired cluster number to display the plots that belong to that cluster. Slide back to 20 to show all.

Cluster #: 17

**Filter by Text:**

Search keyword to filter out the plot. It will search abstracts, titles, journal and authors. Please keep in mind that only 150 words of abstracts, and 10 authors are kept in the plot to reduce the size. Press enter when ready. C and press enter to reset the plot.

Search:

Keywords: probe, dna, primer, target, respiratory, clinical, rt-pcr, surface, device, antibody, cell, species, identification, organism, viral, extraction, strain, den, dengue, sputum, tgev, rt-pcr, extract, hmpv, air, urine, subtype, rsv, mngs, sampler, saliva, bvdv, tile, filter, blood, ebv, type, swine, bovine, hors, fmd, gene, animal, baumannii, chamber, equine, mabs, sequence, specimen, ma

Click on a plot to see the link to the article.

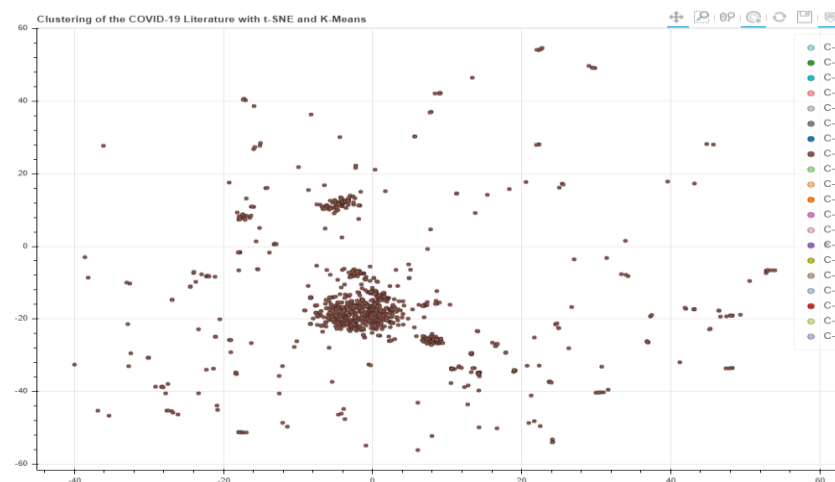


Fig 5.4 slider

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 is 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.

CHAPTER 6

CONCLUSION

It is difficult for health practitioners to keep up with new research on COVID-19 due to the vast amount of literature and the virus's rapid spread. Similar research papers are grouped together to make finding related publications easier. A scatter plot can be used to display a selection of literature by using clustering for labelling and dimensionality reduction for visualisation. Topic modelling will be used to find the keywords of each cluster in order to find significance in the clusters.

Users would be able to view the plot in its entirety or filter the data by cluster. If a more specific scope is needed, the plot will have a search feature that will restrict the production to only papers that include the search word. When you hover your mouse over a point on the storey, you'll see basic details like the title, author, journal, and abstract. When you click on a point, a menu will appear with a URL that can be used to access the entire publication.

Health care staff, sanitation workers, and many other important personnel are out there keeping the world afloat during this tough period. This platform was developed to make it easier for qualified professionals to search through many, many virus-related publications and find their own results. If there will be an app developed in the future specifically for COVID, this optimization technique will list out links to several articles regarding risk factors, precautions to be taken, vaccine side effects, allergic reactions, efficiency of the vaccine etc.

REFERENCE

- [6]A. Gunta and R. Dubey, "An Improved Document Clustering Approach with Multi-Viewpoint Based on Different Similarity Measures," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, pp. 152-157, doi: 10.1109/ICCONS.2018.8663209.
- [7]K. M. Hammouda and M. S. Kamel, "Efficient phrase-based document indexing for Web document clustering," in IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 10, pp. 1279-1296, Oct. 2004, doi: 10.1109/TKDE.2004.58.
- [8]M. Rafi, M. Maujood, Murtaza Munawar Fazal and Syed Muhammad Ali, "A comparison of two suffix tree-based document clustering algorithms," 2010 International Conference on Information and Emerging Technologies, Karachi, Pakistan, 2010, pp. 1-5, doi: 10.1109/ICIET.2010.5625688.
- [9]N. Abd Rahman, Z. Abu Bakar and N. S. S. Zulkefli, "Malay document clustering using complete linkage clustering technique with Cosine Coefficient," 2015 IEEE Conference on Open Systems (ICOS), Melaka, Malaysia, 2015, pp. 103-107, doi: 10.1109/ICOS.2015.7377286.
- [1][http://www.ijiere.com/\(X\(1\)S\(ra4gmapne2vzi5zd1qtkcr4g\)\)/FinalPaper/FinalPaper20141214105543896.pdf](http://www.ijiere.com/(X(1)S(ra4gmapne2vzi5zd1qtkcr4g))/FinalPaper/FinalPaper20141214105543896.pdf)
- [2] <http://www.jatit.org/volumes/Vol98No22/3Vol98No22.pdf>
- [3]https://www.researchgate.net/publication/335700827_Survey_on_Semantic_Similarity_Based_on_Document_Clustering
- [4] <https://www.ijsr.net/archive/v3i11/T0NUMTQxMjk1.pdf>
- [5]<https://towardsdatascience.com/clustering-documents-with-python-97314ad6a78d>

APPENDIX

A. CODE SOURCE

	5th International Conference on Computing Methodologies and Communication (ICCMC 2021) 08-10, April 2021 iccmc.com/2021/ info.iccmc@gmail.com				
<u>Letter of Acceptance</u>					
To Ritika Catherine. G, Adini Hemanjali, UG Scholar, Department of Computer Science and Engineering, Sathyabama Institute of Science and Technology Chennai, India. S Revathy, Associate Professor, V. Maria Ann, Associate Professor, L. Mary Gladence, Associate Professor, P. Jeyanthi, Associate Professor, School of Computing, Sathyabama Institute of Science and Technology Chennai, India. Greetings!! On behalf of the Surya Engineering College Erode, we take great pleasure in inviting you to attend the “5th International Conference on Computing Methodologies and Communication (ICCMC 2021)” which is going to be held during 08-10, April 2021 at Erode, India. We are glad to inform you that your submitted article entitled “Document Clustering on COVID literature using Machine Learning” [Paper id: ICCMC378] has been accepted for publication at ICCMC 2021. We welcome you to join us and share your research and views. This conference will offer you an unforgettable experience in exploring new opportunities. We look forward to seeing you at Erode, India. With Thanks, Yours' Sincerely  Dr. E. Baraneetharan Associate Professor & Head, Dept. of EEE, Surya Engineering College, Erode, India.					
Proceedings by 	Previous ICCMC Publications <table border="1"><tr><td>ICCMC 2020 Publication</td></tr><tr><td>ICCMC 2019 Publication</td></tr><tr><td>ICCMC 2018 Publication</td></tr><tr><td>ICCMC 2017 Publication</td></tr></table>	ICCMC 2020 Publication	ICCMC 2019 Publication	ICCMC 2018 Publication	ICCMC 2017 Publication
ICCMC 2020 Publication					
ICCMC 2019 Publication					
ICCMC 2018 Publication					
ICCMC 2017 Publication					

REFERENCES

- [11] A. Gupta and R. Dubey, "An Improved Document Clustering Approach with Multi-Viewpoint Based on Different Similarity Measures," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Shadara, India, 2018, pp. 152-157, doi: 10.1109/ICICCS.2018.8663209.
- [12] K. M. Hameedy and M. S. Kamel, "Efficient phrase-based document indexing for Web document clustering," in IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 10, pp. 1279-1290, Oct. 2004, doi: 10.1109/TKDE.2004.58.
- [13] M. Rafi, M. Maqsood, Mariyza Munawar Faraz and Syed Muhammad Ali, "A comparison of two suffix tree-based document clustering algorithms," 2010 International Conference on Information and Emerging Technologies, Karachi, Pakistan, 2010, pp. 1-5, doi: 10.1109/CIET.2010.5625688.
- [14] N. Abd Rahman, Z. Abu Bakar and N. S. S. Zulkefli, "Malay document clustering using complete linkage clustering technique with Cosine Coefficient," 2015 IEEE Conference on Open Systems (ICOS), Melaka, Malaysia, 2015, pp. 103-107, doi: 10.1109/ICOS.2015.7377286.
- [15] N. Shah and S. Mahajan, "Scalability analysis of semantics based distributed document clustering algorithms," 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kerala, India, 2017, pp. 763-768, doi: 10.1109/ICICICT.2017.8342660.
- [16] R. Agrawal and M. Phatak, "A novel algorithm for automatic document clustering," 2013 3rd IEEE International Advanced Computing Conference (IACC), Ghaziabad, India, 2013, pp. 877-882, doi: 10.1109/IACC.2013.6514342.
- [17] R. K. Mishra, R. Sani and S. Bagri, "Text document clustering on the basis of inter passage approach by using K-means," International Conference on Computing, Communication & Automation, Greater Noida, India, 2015, pp. 110-113, doi: 10.1109/CCA.2015.7148354.
- [18] S. Kumar, M. Kathuria, A. K. Gupta and M. Rani, "Fuzzy clustering of web documents using equivalence relations and fuzzy hierarchical clustering," 2012 CSI Sixth International Conference on Software Engineering (CONSEG), Indore, India, 2012, pp. 1-5, doi: 10.1109/CONSEG.2012.6349498.
- [19] S. Park, D. U. An and C. I. Choo, "Document Clustering Method Using Weighted Semantic Features and Cluster Similarity," 2010 Third IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning, Kaohsiung, Taiwan, 2010, pp. 185-187, doi: 10.1109/DIGTEL.2010.23.
- [10] S. Xu, Z. Lu and G. Gu, "A Fast Spectral Method to Solve Document Cluster Ensemble Problem," 2008 International Multi-symposium on Computer and Computational Sciences, Shanghai, China, 2008, pp. 180-183, doi: 10.1109/IMSCCS.2008.8.

COVID 19 DOC CLUSTERING-1

ORIGINALITY REPORT

11%

SIMILARITY INDEX

5%

INTERNET SOURCES

11%

PUBLICATIONS

6%

STUDENT PAPERS

```
from IPython.display import Image
Image(filename='/kaggle/input/kaggle-resources/bokeh_plot.png', width=800, height=800)
```

```
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
plt.style.use('ggplot')
```

```
root_path = '/kaggle/input/CORD-19-research-challenge/'
metadata_path = f'{root_path}/metadata.csv'
meta_df = pd.read_csv(metadata_path, dtype={
    'pubmed_id': str,
    'Microsoft Academic Paper ID': str,
    'doi': str
})
meta_df.head()
```

```
meta_df.info()
```

```
all_json = glob.glob(f'{root_path}/**/*.json', recursive=True)
len(all_json)
```

```
def __init__(self, file_path):
    with open(file_path) as file:
        content = json.load(file)
        self.paper_id = content['paper_id']
        self.abstract = []
        self.body_text = []
        # Abstract
        for entry in content['abstract']:
            self.abstract.append(entry['text'])
        # Body text
        for entry in content['body_text']:
            self.body_text.append(entry['text'])
        self.abstract = '\n'.join(self.abstract)
        self.body_text = '\n'.join(self.body_text)
    def __repr__(self):
        return f'{self.paper_id}: {self.abstract[:200]}... {self.body_text[:200]}...'
first_row = FileReader(all_json[0])
print(first_row)
```

```
def get_breaks(content, length):
    data = ""
    words = content.split(' ')
    total_chars = 0

    # add break every length characters
    for i in range(len(words)):
        total_chars += len(words[i])
        if total_chars > length:
            data = data + "<br>" + words[i]
            total_chars = 0
        else:
            data = data + " " + words[i]
    return data
```

```
dict_ = {'paper_id': [], 'doi': [], 'abstract': [], 'body_text': [], 'authors': [], 'title': [], 'journal': [], 'abstract_summary': []}
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(' ')[0:100]
    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)

# get metadata information
meta_data = meta_df.loc[meta_df['sha'] == content.paper_id]

try:
    # if more than one author
    authors = meta_data['authors'].values[0].split(';')
    if len(authors) > 2:
        # if more than 2 authors, take them all with html tag breaks in between
        dict_['authors'].append(get_breaks(' '.join(authors), 40))
    else:
        # authors will fit in plot
        dict_['authors'].append(" ".join(authors))
except Exception as e:
    # if only one author - or Null value
    dict_['authors'].append(meta_data['authors'].values[0])

# add the title information, add breaks when needed
try:
    title = get_breaks(meta_data['title'].values[0], 40)
    dict_['title'].append(title)
# if title was not provided
except Exception as e:
    dict_['title'].append(meta_data['title'].values[0])

# add the journal information
dict_['journal'].append(meta_data['journal'].values[0])

# add doi
dict_['doi'].append(meta_data['doi'].values[0])

df_covid = pd.DataFrame(dict_, columns=['paper_id', 'doi', 'abstract', 'body_text', 'authors', 'title',
    'journal', 'abstract_summary'])
df_covid.head()

df_covid['abstract_word_count'] = df_covid['abstract'].apply(lambda x: len(x.strip().split())) # word count in abstract

```

```

df_covid['body_word_count'] = df_covid['body_text'].apply(lambda x: len(x.strip().split())) # word count in body
df_covid['body_unique_words']=df_covid['body_text'].apply(lambda x:len(set(str(x).split())) # number of unique words in body
df_covid.head()

df_covid.info()

df_covid['abstract'].describe(include='all')

df_covid.drop_duplicates(['abstract', 'body_text'], inplace=True)
df_covid['abstract'].describe(include='all')

df_covid['body_text'].describe(include='all')

df_covid.head()

df_covid.describe()

df = df_covid.sample(10000, random_state=42)
del df_covid

df.dropna(inplace=True)
df.info()

from tqdm import tqdm
from langdetect import detect
from langdetect import DetectorFactory

DetectorFactory.seed = 0

languages = []

for ii in tqdm(range(0,len(df))):
    text = df.iloc[ii]['body_text'].split(" ")

    lang = "en"
    try:
        if len(text) > 50:
            lang = detect(" ".join(text[:50]))
        elif len(text) > 0:
            lang = detect(" ".join(text[:len(text)]))

    except Exception as e:
        all_words = set(text)
        try:
            lang = detect(" ".join(all_words))
        except Exception as e:

            try:
                lang = detect(df.iloc[ii]['abstract_summary'])
            except Exception as e:
                lang = "unknown"
            pass
        languages.append(lang)

from pprint import pprint

```

```

languages_dict = {}
for lang in set(languages):
    languages_dict[lang] = languages.count(lang)

print("Total: {}".format(len(languages)))
pprint(languages_dict)

df['language'] = languages
plt.bar(range(len(languages_dict)), list(languages_dict.values()), align='center')
plt.xticks(range(len(languages_dict)), list(languages_dict.keys()))
plt.title("Distribution of Languages in Dataset")
plt.show()

df = df[df['language'] == 'en']
df.info()

from IPython.utils import io
with io.capture_output() as captured:
    !pip install https://s3-us-west-2.amazonaws.com/ai2-s2-scispace/releases/v0.2.4/en_core_sci_lg-0.2.4.tar.gz

import spacy
from spacy.lang.en.stop_words import STOP_WORDS

import en_core_sci_lg

import string

punctuations = string.punctuation
stopwords = list(STOP_WORDS)
stopwords[:10]

custom_stop_words = [
    'doi', 'preprint', 'copyright', 'peer', 'reviewed', 'org', 'https', 'et', 'al', 'author', 'figure',
    'rights', 'reserved', 'permission', 'used', 'using', 'biorxiv', 'medrxiv', 'license', 'fig', 'fig.',
    'al.', 'Elsevier', 'PMC', 'CZI', 'www'
]

for w in custom_stop_words:
    if w not in stopwords:
        stopwords.append(w)

parser = en_core_sci_lg.load(disable=["tagger", "ner"])
parser.max_length = 7000000

def spacy_tokenizer(sentence):
    mytokens = parser(sentence)
    mytokens = [ word.lemma_.lower().strip() if word.lemma_ != "-PRON-" else word.lower_ for word in mytokens ]
    mytokens = [ word for word in mytokens if word not in stopwords and word not in punctuations ]
    mytokens = " ".join([i for i in mytokens])
    return mytokens

tqdm.pandas()
df["processed_text"] = df["body_text"].progress_apply(spacy_tokenizer)

```

```

import seaborn as sns

sns.distplot(df['body_word_count'])
df['body_word_count'].describe()
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
text = df['processed_text'].values
X = vectorize(text, 2 ** 12)
X.shape

from sklearn.decomposition import PCA

pca = PCA(n_components=0.95, random_state=42)
X_reduced= pca.fit_transform(X.toarray())
X_reduced.shape
from sklearn.cluster import KMeans
Image(filename='/kaggle/input/kaggle-resources/kmeans.PNG', width=800, height=800)

from sklearn import metrics
from scipy.spatial.distance import cdist

distortions = []
K = range(2, 50)
for k in K:
    k_means = KMeans(n_clusters=k, random_state=42).fit(X_reduced)
    k_means.fit(X_reduced)
    distortions.append(sum(np.min(cdist(X_reduced, k_means.cluster_centers_, 'euclidean'), axis=1)) / X.shape[0])

X_line = [K[0], K[-1]]
Y_line = [distortions[0], distortions[-1]]

# Plot the elbow
plt.plot(K, distortions, 'b-')
plt.plot(X_line, Y_line, 'r')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal k')
plt.show()

k = 20
kmeans = KMeans(n_clusters=k, random_state=42)
y_pred = kmeans.fit_predict(X_reduced)
df['y'] = y_pred
from sklearn.manifold import TSNE

tsne = TSNE(verbose=1, perplexity=100, random_state=42)
X_embedded = tsne.fit_transform(X.toarray())

from matplotlib import pyplot as plt
import seaborn as sns

```

```

sns.set(rc={'figure.figsize':(15,15)})

palette = sns.color_palette("bright", 1)

sns.scatterplot(X_embedded[:,0], X_embedded[:,1], palette=palette)
plt.title('t-SNE with no Labels')
plt.savefig("t-sne_covid19.png")
plt.show()

%matplotlib inline
from matplotlib import pyplot as plt
import seaborn as sns
sns.set(rc={'figure.figsize':(15,15)})

palette = sns.hls_palette(20, l=.4, s=.9)
sns.scatterplot(X_embedded[:,0], X_embedded[:,1], hue=y_pred, legend='full', palette=palette)
plt.title('t-SNE with Kmeans Labels')
plt.savefig("improved_cluster_tsne.png")
plt.show()

from sklearn.decomposition import LatentDirichletAllocation
from sklearn.feature_extraction.text import CountVectorizer
Image(filename='lda.jpg', width=600, height=600)

vectorizers = []

for ii in range(0, 20):
    vectorizers.append(CountVectorizer(min_df=5, max_df=0.9, stop_words='english', lowercase=True, token_pattern='[a-zA-Z\\-\\_]{2,}'))

vectorizers[0]

vectorized_data = []

for current_cluster, cvec in enumerate(vectorizers):
    try:
        vectorized_data.append(cvec.fit_transform(df.loc[df['y'] == current_cluster, 'processed_text']))
    except Exception as e:
        print("Not enough instances in cluster: " + str(current_cluster))
        vectorized_data.append(None)
len(vectorized_data)

NUM_TOPICS_PER_CLUSTER = 20

lda_models = []
for ii in range(0, 20):
    lda = LatentDirichletAllocation(n_components=NUM_TOPICS_PER_CLUSTER, max_iter=10, learning_method='online', verbose=False, random_state=42)
    lda_models.append(lda)

lda_models[0]

def selected_topics(model, vectorizer, top_n=3):
    current_words = []
    keywords = []

```

```

for idx, topic in enumerate(model.components_):
    words = [(vectorizer.get_feature_names()[i], topic[i]) for i in topic.argsort()[: -top_n - 1:-1]]
    for word in words:
        if word[0] not in current_words:
            keywords.append(word)
            current_words.append(word[0])

keywords.sort(key = lambda x: x[1])
keywords.reverse()
return_values = []
for ii in keywords:
    return_values.append(ii[0])
return return_values

all_keywords = []
for current_vectorizer, lda in enumerate(lda_models):

    if vectorized_data[current_vectorizer] != None:
        all_keywords.append(selected_topics(lda, vectorizers[current_vectorizer]))

all_keywords[0][:10]
len(all_keywords)

f=open('topics.txt','w')

count = 0

for ii in all_keywords:

    if vectorized_data[count] != None:
        f.write(', '.join(ii) + "\n")
    else:
        f.write("Not enough instances to be determined. \n")
        f.write(', '.join(ii) + "\n")
    count += 1

f.close()

import pickle
pickle.dump(df, open("df_covid.p", "wb" ))
pickle.dump(X_embedded, open("X_embedded.p", "wb" ))
pickle.dump(y_pred, open("y_pred.p", "wb" ))

def classification_report(model_name, test, pred):
    from sklearn.metrics import precision_score, recall_score
    from sklearn.metrics import accuracy_score
    from sklearn.metrics import f1_score

    print(model_name, ":\n")
    print("Accuracy Score: ", '{:,.3f}'.format(float(accuracy_score(test, pred)) * 100), "%")
    print("    Precision: ", '{:,.3f}'.format(float(precision_score(test, pred, average='macro')) * 100
), "%")
    print("    Recall: ", '{:,.3f}'.format(float(recall_score(test, pred, average='macro')) * 100), "%")
    print("    F1 score: ", '{:,.3f}'.format(float(f1_score(test, pred, average='macro')) * 100), "%")

```

```

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X.toarray(), y_pred, test_size=0.2, random_state=42)

print("X_train size:", len(X_train))
print("X_test size:", len(X_test), "\n")

from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_val_predict
from sklearn.linear_model import SGDClassifier
sgd_clf = SGDClassifier(max_iter=10000, tol=1e-3, random_state=42, n_jobs=4)
sgd_clf.fit(X_train, y_train)
sgd_pred = cross_val_predict(sgd_clf, X_train, y_train, cv=3, n_jobs=4)
classification_report("Stochastic Gradient Descent Report (Training Set)", y_train, sgd_pred)

sgd_pred = cross_val_predict(sgd_clf, X_test, y_test, cv=3, n_jobs=4)

classification_report("Stochastic Gradient Descent Report (Training Set)", y_test, sgd_pred)

sgd_cv_score = cross_val_score(sgd_clf, X.toarray(), y_pred, cv=10)
print("Mean cv Score - SGD: {:.3f}".format(float(sgd_cv_score.mean()) * 100), "%")

import os
main_path = os.getcwd()
lib_path = '/kaggle/input/kaggle-resources'
os.chdir(lib_path)

from call_backs import input_callback, selected_code # file with customJS callbacks for bokeh
import bokeh
from bokeh.models import ColumnDataSource, HoverTool, LinearColorMapper, CustomJS, Slider, TapTool, TextInput
from bokeh.palettes import Category20
from bokeh.transform import linear_cmap, transform
from bokeh.io import output_file, show, output_notebook
from bokeh.plotting import figure
from bokeh.models import RadioButtonGroup, TextInput, Div, Paragraph
from bokeh.layouts import column, widgetbox, row, layout
from bokeh.layouts import column

os.chdir(main_path)

import os

topic_path = 'topics.txt'
with open(topic_path) as f:
    topics = f.readlines()

output_notebook()
y_labels = y_pred
source = ColumnDataSource(data=dict(
    x= X_embedded[:,0],
    y= X_embedded[:,1],
    x_backup = X_embedded[:,0],
    y_backup = X_embedded[:,1],
    desc= y_labels,
    titles= df['title'],
    authors = df['authors'],

```

```

journal = df['journal'],
abstract = df['abstract_summary'],
labels = ["C-" + str(x) for x in y_labels],
links = df['doi']
))

hover = HoverTool(tooltips=[
    ("Title", "@titles{safe}"),
    ("Author(s)", "@authors{safe}"),
    ("Journal", "@journal"),
    ("Abstract", "@abstract{safe}"),
    ("Link", "@links")
],
point_policy="follow_mouse")

mapper = linear_cmap(field_name='desc',
                    palette=Category20[20],
                    low=min(y_labels), high=max(y_labels))
plot = figure(plot_width=1200, plot_height=850,
              tools=[hover, 'pan', 'wheel_zoom', 'box_zoom', 'reset', 'save', 'tap'],
              title="Clustering of the COVID-19 Literature with t-SNE and K-Means",
              toolbar_location="above")
plot.scatter('x', 'y', size=5,
            source=source,
            fill_color=mapper,
            line_alpha=0.3,
            line_color="black",
            legend = 'labels')
plot.legend.background_fill_alpha = 0.6

text_banner = Paragraph(text= 'Keywords: Slide to specific cluster to see the keywords.', height=45)
input_callback_1 = input_callback(plot, source, text_banner, topics)
div_curr = Div(text="Click on a plot to see the link to the article.", height=150)
callback_selected = CustomJS(args=dict(source=source, current_selection=div_curr), code=selected_code())
taptool = plot.select(type=TapTool)
taptool.callback = callback_selected
slider = Slider(start=0, end=20, value=20, step=1, title="Cluster #", callback=input_callback_1)
keyword = TextInput(title="Search:", callback=input_callback_1)
input_callback_1.args["text"] = keyword
input_callback_1.args["slider"] = slider

slider.sizing_mode = "stretch_width"
slider.margin=15

keyword.sizing_mode = "scale_both"
keyword.margin=15

div_curr.style={'color': '#BF0A30', 'font-family': 'Helvetica Neue, Helvetica, Arial, sans-serif;', 'font-size': '1.1em'}
div_curr.sizing_mode = "scale_both"
div_curr.margin = 20

```

```

text_banner.style={'color': '#0269A4', 'font-family': 'Helvetica Neue, Helvetica, Arial, sans-serif;'
, 'font-size': '1.1em'}
text_banner.sizing_mode = "scale_both"
text_banner.margin = 20

plot.sizing_mode = "scale_both"
plot.margin = 5

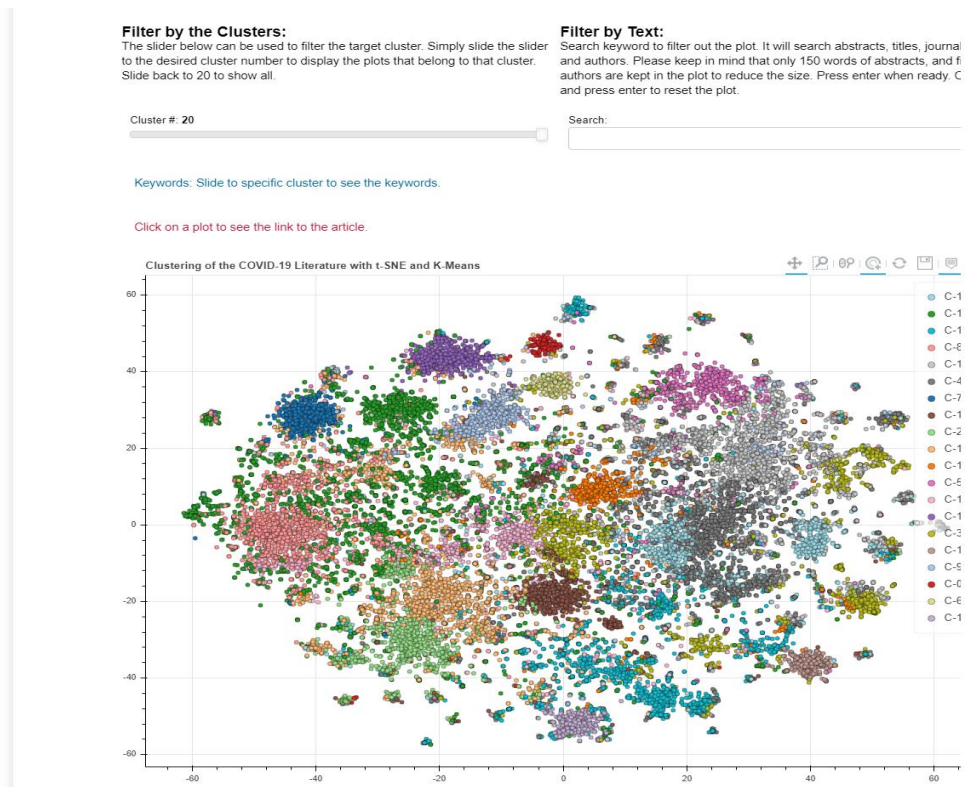
r = row(div_curr, text_banner)
r.sizing_mode = "stretch_width"

l = layout([
    [slider, keyword],
    [text_banner],
    [div_curr],
    [plot],
])
l.sizing_mode = "scale_both"

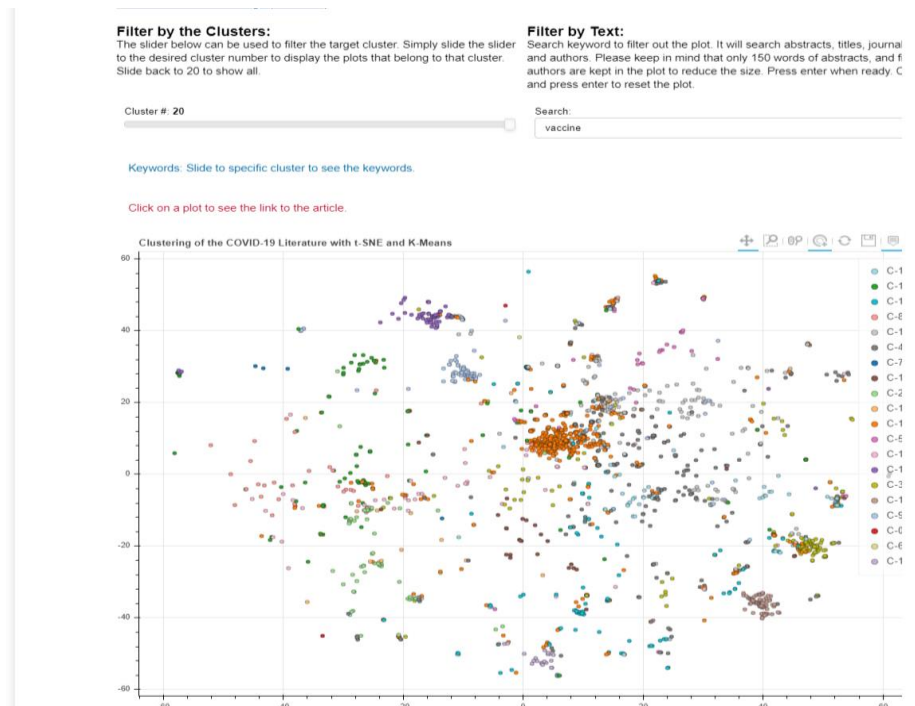
# show
output_file('t-sne_covid-19_interactive.html')
show(l)

```

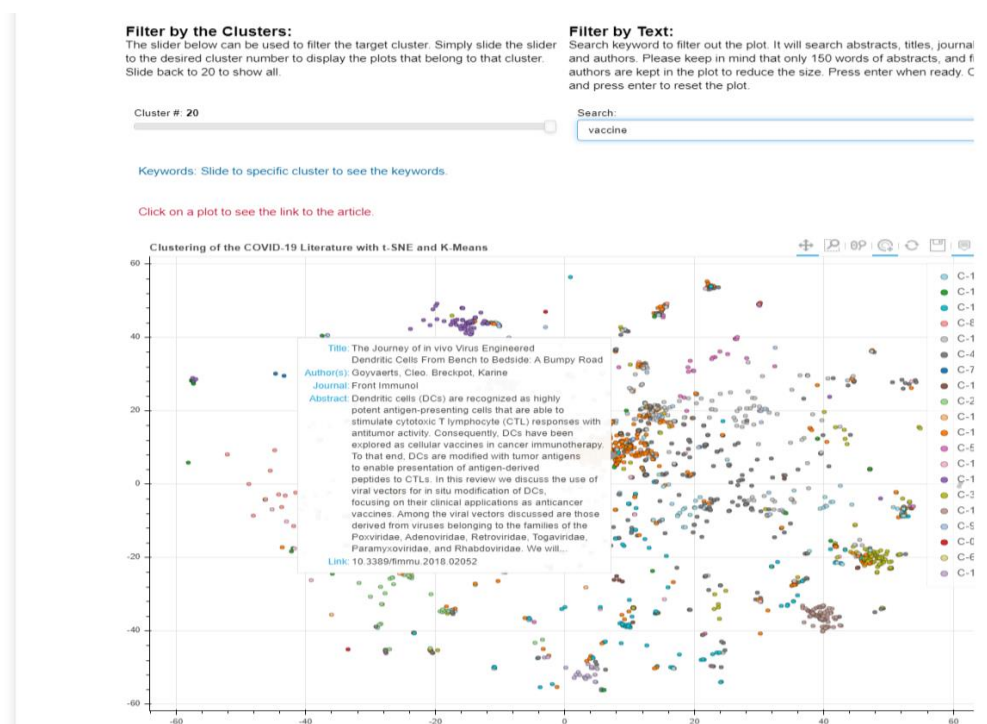
B.SCREEN SHOTS



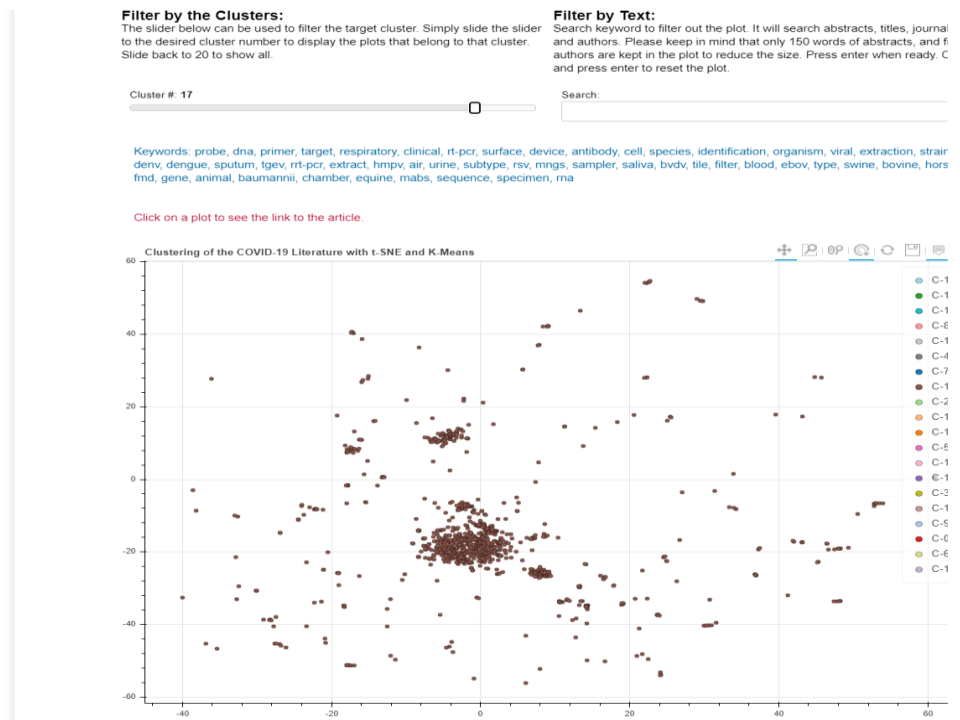
In the search tool, when you search for a keyword, datapoints or clusters that belong to that topic or keyword will appear. Here, we have searched for the term “vaccine”.



In the below figure, the slider is used to display a particular cluster. For example, when you bring the slider to 17, cluster with the label C-17 in burgundy colour is displayed.



In this below figure, when you bring the cursor and hover over the datapoints, you get to see the descriptive information about a singular article.



C.JOURNAL PAPER

Abstract— Coronavirus is a worldwide pandemic, which started in Wuhan, China, in December, has extended to contact every edge of the globe. Many individuals around the planet have been sick from the virus, and countless others have passed away. On January 11, 2020, China reported the first case of COVID death. Before long, the infection had begun to spread all around the world. There are about 116 million cases, and the population is yet to be treated. The study and progress in the research of this virus have reduced researchers' ability to get relevant information over the internet. As mutations of infections and microbes are inescapable, a danger for such pandemics remains. To guarantee that the harm is done is lesser, a literature clustering technique can be used to group relevant information together, making accessibility easier for researchers and enthusiasts alike. Furthermore, by using k-means clustering for labeling the clusters and Principal Component Analysis for dimensionality reduction, t-SNE for better visualization, a scatter plot may be used to represent the grouped collection of literature. On this bokeh plot, articles that are similar share a tag, whereas articles that are not similar are placed so far apart. Latent Dirichlet Allocation will be used to extract keywords from each cluster in order to find credibility in the groups.

Keywords—coronavirus, researchers, articles, clustering, topics

II. INTRODUCTION

In late December 2019, China health professionals reported cases of pneumonia in Wuhan. A few days later, a study on the history and origin of the novel coronavirus was released and made public for health professionals. SARS-CoV-2 was the name given to the new coronavirus by the researchers. The Coronaviridae family includes SARS-CoV-2 and other related diseases. Since the latest strain of the virus was discovered, it has gone global, killing millions of people in just couple of days. Coronaviruses are a family of different that has been existed for around three decades, and they cause respiratory illness. A novel coronavirus (novel CoV) is a new virus mutant that was not found identified in the

human race, and they have been termed as "COVID-19 virus". By that time, over 118k cases had been reported in 114 countries, with over 4k fatalities. The WHO European Zone had become the pandemic's central focus by mid-March 2020, documenting over 40 percent of all known cases globally. Starting on April 28, 63 percent of the worldwide mortality from the infection was from the Region. Since the primary cases were accounted for, WHO has worked nonstop to help nations plan and react to the COVID-19 pandemic. The White House, in association with leading research groups, has published the CORD-19 dataset, which features papers in an unstructured manner, in reaction to the disease outbreak. SARS-CoV-2 and other coronaviruses are featured prominently in the reports. Over 400k scholarly publications are included in the CORD-19 dataset, with over 150k of them having complete transcript. It is permissible to use natural language processing algorithms and other machine learning techniques to gather insights that will aid in the ongoing struggle against the imminent infection. Machine Learning is a branch of Artificial Intelligence that self-learns by trial and error. If a computer program's success at tasks in T , as measured by P , increases with experience E , it is said to learn from experience E about certain class of tasks T and performance measure P , as Tom Mitchel once said. Machine Learning performs tasks without being told to do so. They have two objectives: to classify data on models and predict possible outcomes using these models. Probability plays a major part in machine learning in terms of decisions and inferences from samples or populations. There are many sub-fields within machine learning to study different types of data. One of those subfields is Natural Language Processing (NLP). NLP can read, analyze, and manipulate data in general English language (understandable). NLP analyses these English text documents improves them by turning the unstructured data into structured data. Some of the most common NLP algorithms used are Word clouds, topic modeling, principal component analysis (used for dimensionality reduction), etc.

III. LITERATURE REVIEW

K.M. Hammouda et al. [2] presents two essential elements of effective document clustering. The document index graph is a novel phrase-based document index model that allows for progressive creation of a phrase-based index of the document set with a focus on potency rather than relying solely on single-term indices. The incremental document clustering algorithmic programme in the second half supports optimising cluster tightness by meticulously observing the pair-wise document similarity distribution inside clusters. When those two sections are merged, an associate is created in the underlying model for reliable and accurate document similarity calculation, resulting in significantly improved web document clustering results.

Sen Xu et al. [10] introduces a spectral strategy for fixing disputes with paper cluster ensembles. Since spectral clustering must eventually decipher a matrix's eigenvalues and eigenvectors, it's computationally untamed for large text datasets. They obtain a feasible algorithmic programme by applying algebraic transformation to the similarity matrix.

Sun Park et al. [9] explain how a database clustering approach based on weighted semantic features and cluster similarity is used to cluster meaningful topics from a document collection. The proposed approach would boost the quality of document clustering by avoiding clustering documents with high topic similarities but no context between cluster and document using weighted semantic features. Furthermore, it employs cluster similarity to eliminate the differences between documents in clusters and prevent NMF's biased inherent semantics expressed in clusters. The proposed method outclasses other document clustering methods, according to the findings of the experiments.

Muhammad Rafi et al. [3] explains the suffix tree data model was helped by comparing and introducing distinct approaches to document clustering. The first would be Efficient Phrase-based Document Clustering, which extracts phrases from documents to construct compact document illustrations and clusters the documents using a similarity-based common suffix tree. The second approach employs a common phrase, which is a sequence-based document clustering method that extracts the document's popular sequence of words. It performs the compact illustration using the regular pattern signified array, and then clusters the compressed documents using the document clustering approach. These algorithms use collective hierarchic document clustering to perform the specific clustering step; the differences between these techniques are the predominant supported phrase abstraction, model illustration as a compact document, and the similarity measures used for clustering.

S. Kumar et al. [8] explains using equivalence relations and fuzzy hierarchical clustering, an algorithm and experimental results for fuzzy clustering of web documents are presented.

R. Agrawal et al. [6] have created a learning method that constructs several clusters for every unknown text

dataset and correctly clusters the files based on Cosine Similarity.

R. K. Mishra et al. [7] presents an alternative inter-passage dependent clustering strategy that can cluster portions of documents based on the similarity. The data will be a set of documents containing multi-topic segments gathered from the internet. The section score of the sections of the reports is computed by SentiWordNet. On the intra-document level, section-based clustering will be implemented based on the section score.

N. Abd Rahman et al. [4] concentrates on document clustering for the compilation of Malay texts. It contains 2028 Hadith documents from the journal Sahih Bukhari that have been adapted into Malay. The results of using the Complete Linkage Clustering algorithm with the Cosine Coefficient on Malay translated Hadith documents are presented in their paper.

N. Shah et al. [5] showed The theory of linguistics is expected to be mainly based on distributed document clustering using Hadoop and MapReduce. The distributed version of two well-known algorithms, K-Means and Bisecting K-Means, is executed; testing and contrasting both of these algorithms on single-node and multi-node Hadoop clusters for comprehensiveness and consistency is done.

Anjali Gunta et al. [1] introduces two linked document clustering strategies and document clustering based on Multiview point-based similarity nature. The current clustering dissimilarity or similarity live document only uses one viewpoint: the source, indicating how it only uses one point of comparison. They, on the other hand, employ a range of references.

IV. PROPOSED WORK

It's difficult for medical professionals to stay updated with the latest findings about COVID-19 given the large number of scientific articles and the virus's exponential growth. This study represents an interactive scatter plot of documents associated with COVID-19, in which information related is plotted in the same cluster. Experts can quickly spot material related to a central concept by grouping the literature in this way. Each publication is linked to a larger topic cluster, which eliminates the need to explicitly search for related work.

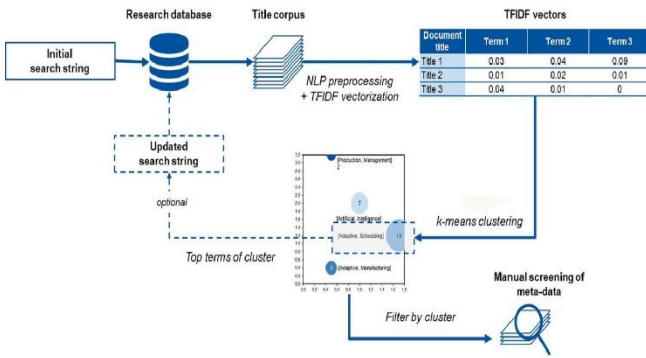


Fig 1: System Architecture

A. Loading

Firstly, we start by uploading the csv file, which is the metadata. Then we upload and read the zip file that contain all the JSON file which has the full text information of the articles. With the help of helper function, the JSON files can be read.

B. Data Pre-processing

With pre-processing the data, we are adding new columns. They are word counts for abstract, for body text and for count of unique words used in body text. Set() is used to convert the iterations of words to distinct or unique value. Part of pre-processing also includes handling duplicates. Using Describe() function, we come to know that majority of the body text or abstract are unique yet there are no duplicate. This is the case of missing abstract and body text. We take a sample of 10k papers, from those papers, we drop 699 article that don't provide abstract or body text. This is done to make it easier to cluster using k-mean algorithm. Then we drop articles that are not in English, since translating those articles to English will lose its real semantic meaning. Then we remove the stop words that are not important but are constantly repeated. Then we convert texts to lowercase and remove the punctuation.

C. Vectorization

The next step is to vectorize the text. From all the vectorizers, we will be use TF-IDF, also know as Term Frequency Inverse Document Frequency. This vectorizer parses text and returns a float value as a vector that indicates the frequency of a word in that document or all of documents. Since an article has from 5000 to 20,000 words, it's a lot to parse all of the words. So we will provide top 4096 (2 ** 12). The formula for TF is:

$$TF(t) = \frac{\text{Number of times term } t \text{ appears in the document}}{\text{Total number of terms in the document}}$$

The formula for Inverse Document frequency is:

$$IDF(t) = \log \left(\frac{\text{Total amount of documents}}{\text{Number of documents with the term } t \text{ in it}} \right)$$

Both TF and IDF multiplied to give a vector and determines the frequency and importance of the term t.

D. Dimensionality Reduction

Principle Component Analysis (PCA) is an algorithm that is used for dimensionality reduction. It reduces the multi-dimensional to two-dimensional. It keeps majority of the data while removing the outliers without losing its semantic meaning. The reduced data is used by k-means. K-means is applied where a datapoint is made as the centroid and mean distance is taken and clustered. The centroids are incrementally modified.

But before applying k-means, we need to find the optimal k value to determine right number of clusters give the most accurate results. For that, we use the elbow method. Distortion is calculated and plotted against k-values. An elbow is formed, that represents around which number of clusters gives accurate results. For simplicity we will be taking 20 as our k value. Then we train k-mean algorithm and predict labels for our cluster and save them in the object 'y_preds'.

E. Clustering and Classifying labels

We will be using t-SNE also known as t-Distributed Stochastic Neighbour Embedding is not only used for dimensionality reduction but also does a great job at clustering and visualization of the plot. Data before PCA is used and reduced and plotted. t-SNE places similar articles together in a cluster and dissimilar articles further away from others. A plot is presents but it is bland. The distinguish between them, using predicted labels by k-means, we set palette and hue as y-preds.

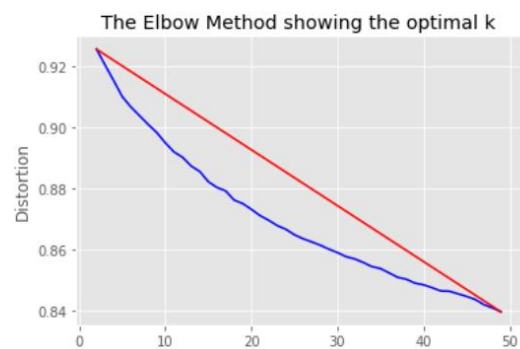


Fig 2: Elbow Method

F. Topic Modelling

Topic modelling is something we do to put on a label on a cluster. For topic modelling, we will be using Latent Dirichlet Allocation (LDA). It is a model that assigns a topic based on a group of documents, with a bunch of related keywords. For each iteration or cluster, we will

vectorize them and LDA model will be applied and it extracts keywords. We will then plot the clusters using bokeh plot. the slider is used to pick a certain cluster and search bar to search for a particular keyword.

G. Proposed Algorithm:

Step 1: Load the metadata of the dataset (.csv). Get the path to all json files and read the files using Helper function

Step 2: Moving to feature engineering, create a column for abstract word count and done for body text word count. Analyse the dataset. Handle duplicates. Looking at the dataset there are many unique values as well as duplicates. Remove the duplicates

Step 3: Then we drop the null values that are present in abstract or body text column. Drop rows that are not in English language. The reason is because when translating the non English articles, it loses its true semantic meaning. Remove the stop words that are repeated yet not important. Process the body text and will transform text to lower case, remove punctuation.

Step 4: Then we move to vectorization. Using TF-IDF, convert the words into vectors. The vector is calculated by how important each term compared to the rest in that particular documents or group of documents.

Step 5: We will apply Principal Component Analysis (PCA) to our vectorized data. It reduces the multi-dimensional to two-dimensional. It keeps majority of the data while removing the outliers without losing its semantic meaning.

Step 6: Run the k-mean algorithm on vectorized text. To find the optimal k value to separate literatures into clusters, we will use a metric called the distance metrics. Through spatial you can perform the elbow method and get optimal k value.

Step 7: t-SNE is mostly used for visualization. It separates articles that are dissimilar to each other and keeps close that are similar. It does a great job at clustering and hence it clusters the articles. The predicted label from k-means saved in 'y_pred' will be set as hue to give colour to the scatter plot, distinguishing the clusters.

Step 8: Moving to topic modelling, we will use LDA (Latent Dirichlet Allocation) where it maps documents to a list of topics by assigning each word in the document to different topics.

Step 9: Create 20 vectorizers, one for each cluster. Vectorize each cluster. Apply LDA model on each cluster. We will now fit all the 20 LDA models. Extract keywords from the clusters and append them to a two-dimensional list.

Step 10: We can plot the output using bokeh interactive plot by importing bokeh module. For making it interactive, create a slider from widgets to view clusters from 0 to 20, a search bar to search for keywords, TapTool to select a particular document. Each cluster are labelled with the label given in y_pred.

Step 11: Show output. You can hover over the plot and each article gives you its author name, publication link, abstract etc.

V. RESULTS AND DISCUSSION

The main aim of this project is clustering the articles related to SARS-CoV-2 and other coronavirus diseases in the last 3 decades, arrange them in a structured way, optimize search results and make retrieving results for the health researchers and doctors easier and faster. The input is a total of 154,598 unstructured data. Then removing unwanted articles, reducing its dimension, clustering the articles, assigning topics and extracting keywords, the data is presented in a bokeh plot.

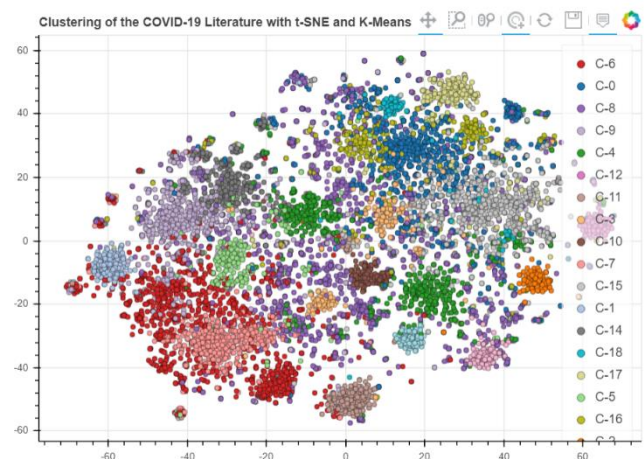


Fig 3: Bokeh Plot

In the search tool, when you search for a keyword, datapoints or clusters that belong to that topic or keyword will appear. Here, we have searched for the term "vaccine".



Fig 4: Plot with keyword search

In the below figure, the slider is used to display a particular cluster. For example, when you bring the slider to 17, cluster with the label C-17 in burgundy colour is displayed.



Fig 5: plot with slider

In this below figure, when you bring the cursor and hover over the datapoints, you get to see the descriptive information about a singular article.



Fig 6: plot with description

VI. CONCLUSION

Articles on some of the key topic can be easily be discovered in close proximity to each other. Example: search the word 'mask' and it will gather articles of that cluster. Providing an organized, structured data peer reviewed will help a health researcher to go through information and articles quickly. If there will be an app developed in the future specifically for COVID, this optimization technique will list out links to several articles regarding risk factors, precautions to be taken, vaccine side effects, allergic reactions, efficiency of the vaccine etc.

References

- [1] A. Gunta and R. Dubey, "An Improved Document Clustering Approach with Multi-Viewpoint Based on Different Similarity Measures," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, pp. 152-157, doi: 10.1109/ICCONS.2018.8663209.
- [2] K. M. Hammouda and M. S. Kamel, "Efficient phrase-based document indexing for Web document clustering," in IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 10, pp. 1279-1296, Oct. 2004, doi: 10.1109/TKDE.2004.58.
- [3] M. Rafi, M. Maujood, Murtaza Munawar Fazal and Syed Muhammad Ali, "A comparison of two suffix tree-based document clustering algorithms," 2010 International Conference on Information and Emerging Technologies, Karachi, Pakistan, 2010, pp. 1-5, doi: 10.1109/ICIET.2010.5625688.
- [4] N. Abd Rahman, Z. Abu Bakar and N. S. S. Zulkefli, "Malay document clustering using complete linkage clustering technique with Cosine Coefficient," 2015 IEEE Conference on Open Systems (ICOS), Melaka, Malaysia, 2015, pp. 103-107, doi: 10.1109/ICOS.2015.7377286.
- [5] N. Shah and S. Mahajan, "Scalability analysis of semantics based distributed document clustering algorithms," 2017 International Conference on Intelligent Computing, Instrumentation and

Control Technologies (ICICT), Kerala, India, 2017, pp. 763-768, doi: 10.1109/ICICT1.2017.8342660.

- [6] R. Agrawal and M. Phatak, "A novel algorithm for automatic document clustering," 2013 3rd IEEE International Advance Computing Conference (IACC), Ghaziabad, India, 2013, pp. 877-882, doi: 10.1109/IAdCC.2013.6514342.
- [7] R. K. Mishra, K. Saini and S. Bagri, "Text document clustering on the basis of inter passage approach by using K-means," International Conference on Computing, Communication & Automation, Greater Noida, India, 2015, pp. 110-113, doi: 10.1109/CCAA.2015.7148354.
- [8] S. Kumar, M. Kathuria, A. K. Gupta and M. Rani, "Fuzzy clustering of web documents using equivalence relations and fuzzy hierarchical clustering," 2012 CSI Sixth International Conference on Software Engineering (CONSEG), Indore, India, 2012, pp. 1-5, doi: 10.1109/CONSEG.2012.6349496.
- [9] S. Park, D. U. An and C. I. Cheon, "Document Clustering Method Using Weighted Semantic Features and Cluster Similarity," 2010 Third IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning, Kaohsiung, Taiwan, 2010, pp. 185-187, doi: 10.1109/DIGITEL.2010.23.
- [10] S. Xu, Z. Lu and G. Gu, "A Fast Spectral Method to Solve Document Cluster Ensemble Problem," 2008 International Multi-symposiums on Computer and Computational Sciences, Shanghai, China, 2008, pp. 180-183, doi: 10.1109/IMSCCS.2008.8.

