Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Seminar Report

On

Web Clustering Engines

By

Prathmesh Wawre

1032190936

Under the guidance of

Prof. Rashmi Rane

MIT-World Peace University (MIT-WPU)

Faculty of Engineering
School of Computer Engineering & Technology

*** 2022-2023 ***

MIT-World Peace University (MIT-WPU)

Faculty of Engineering
School of Computer Engineering & Technology

CERTIFICATE

This is to certify that Mr. Prathmesh Wawre of B.Tech., School of Computer Engineering & Technology, Semester – 9, PRN. No. 1032190936, has successfully completed seminar on

Web Clustering Engines

To my satisfaction and submitted the same during the academic year 2022 - 2023 towards the partial fulfillment of degree of Bachelor of Technology in School of Computer Engineering & Technology under Dr. Vishwanath Karad MIT- World Peace University, Pune.

Prof. Rashmi Rane	Prof. Dr. V. Y .Kulkarni
Seminar Guide	Head

School of Computer Engineering & Technology

List Of Tables

List Of Figures

Acknowledgement

To complete this report work I was fortunate to get advised by the faculties. I wish to convey deep sense of gratitude to all of them.

I am highly indebted to my guide, Prof. Rashmi Rane for their expertise and highly valuable guidance towards the completion of this Report. I would like to special thanks to our head of department Prof. Dr. V. Y. Kulkarni.

I would like to convey special thanks to Prof. Pramod Mundhe for guiding to work on this topic, and assisting in guiding to do a search on the current Web Clustering Engines.

Finally, I would also like to thank my parents, friends and well-wishers for their support, help. Suggestions and encouragement.

Index

1. Abstract

An emerging trend in information retrieval is the use of web clustering engines. They provide a different perspective from the standard search engines' flat ranked list of results by grouping search results by topic. The user might have to filter through a lot of uninteresting items to get the relevant ones because the search results produced by conventional search engines on many subtopics or meanings of a query are mixed together in the list. Search results are divided into several hierarchical groups and clusters by the Web clustering engines, which then display the labels for each cluster. As a result, the user can find the necessary content extremely quickly.

By breaking down the enormous set of search results into smaller clusters, web clustering engines considerably reduce the user's work involved in perusing them. Web clustering engines group search results by topic, providing a different perspective from the flat-ranked list that traditional search engines produce. In this study, we go over the problems that need to be solved while creating a Web clustering engine, such as how to gather and preprocess search results as well as cluster and visualize them.

The data is divided into K clusters using a popular clustering technique based on K-means. In this approach, the cluster centers—a collection of points—are used to identify the groups. The cluster with the closest center contains the data points.

In this seminar we discuss different phases in the implementation of web clustering engines in detail and also incorporate some of the web clustering algorithms, their advantages and issues.

2. Keywords

Web Clustering Engines, Information retrieval, meta search engines, search results clustering, Search results acquisition, Preprocessing, Cluster construction and labeling, Vector Space model, data centric clustering algorithms, description aware algorithms

3. Introduction

3.1 General Introduction

A Web Cluster The systems that cluster the results of web searches are known as engines. This approach organises the search engine's results into a hierarchy of designated clusters (also called categories).

Search engines are a useful resource for finding information on the Internet. They provide a list of results ordered according to how relevant they are to the user's inquiry in response to a query. The user starts at the top of the list and works their way down, looking at each result as they go, until they find the data they are looking for.

While search engines are undoubtedly useful for some search tasks, such locating a company's home page, they could be less useful for answering general or ambiguous queries. The results for several subtopics or interpretations of a query will be grouped together in the list, indicating that the user may need to filter through a lot of uninteresting items to find those that are relevant.

Even commercial systems now use web clustering engines, but the cluster structures they create are still far from ideal for a particular polysemy keyword. The web clustering engines produce more irrelevant clusters even at the single level while excluding only a small number of relevant clusters. In this report, I go into detail regarding the K-Means algorithm as well as the clustering algorithms and how they compare.

3.2 Goal of Clustering Engines

Web clustering engines group search results by subject, providing a different perspective from the flat ranked list that is delivered by traditional search engines.

The cluster hierarchy's main benefits include:

- It creates shortcuts to the elements that pertain to the same meaning. The user can find related pages very easily since web clustering engines aggregate search results with the same meaning into the same cluster. As a result, the search time will be shorter.
- It allows for greater topic comprehension. Web clustering engines are helpful for informational searches in uncharted or dynamic domains since they provide a high-level picture of the query.
- It favors systematic exploration of search results. A clustering engine summarizes the content of many search results in one single view on the first result page, the user may review hundreds of potentially relevant results without the need to download and scroll to subsequent pages.

4. Document Clustering

The technique of discovering subsets of the input's objects (clusters, groups) in such a way that the items within a cluster are similar to one another and the objects from various clusters are unlike from one another is known as clustering. The text-based web document clustering techniques assign each document a personality based on the words (or occasionally phrases) that make up it. Two documents are quite similar if they use a lot of the same words.

The clustering techniques used by the text-based approaches can be used to further categorise them. Additionally, a clustering technique can be either crisp (or hard), which considers non-overlapping partitions, or fuzzy (or soft), with which a document can be classified to more than one cluster. This is dependent on how the algorithm manages uncertainty in terms of cluster overlapping. Since the majority of the current techniques are clear-cut, a document either belongs to a cluster or it doesn't.

4.1 Key Requirements

The following are the key requirements for web document clustering methods:

(1) Relevance: The method ought to produce clusters that group documents relevant to the user’s query.

(2) Browsable Summaries: The user needs to determine at a glance whether a cluster's contents are of interest. Ranked lists of the clusters may infact difficult to browse. Therefore, the method has to provide concise and accurate descriptions of the clusters.

(3) Overlap: Since documents have multiple topics, it is important to avoid confining each document to only one cluster.

(4) Snippet-tolerance: The method ought to produce high quality clusters even when it only has access to the snippets returned by the search engines, as most users are unwilling to wait while the system downloads the original documents off the Web.

(5) Speed: A very patient user might sift through 100 documents in a ranked list presentation. Clustering on the other hand allows the user to browse several related documents. Therefore, the clustering method ought to be able to cluster up to one thousand snippets in a few seconds. For the impatient user, each second counts.

(6) Incrementality: To save time, the method should start to process each snippet as soon as it is received over the Web.

4.2 Types

Based on the relation between the clusters, the clustering algorithm is classified as partitional and hierarchical.

Partitional Clustering Algorithms

K-means, the most popular partitional clustering algorithm, is predicated on the notion that the centroid, or centre, of the cluster, can serve as a reliable representation of the cluster. The first step of the procedure is to choose k cluster centroids. Each document in the collection is then assigned to the cluster with the closest centroid after the cosine distance between each document and the centroids is determined. After all documents have been assigned to clusters, the operation iteratively continues until a predetermined condition is reached. The new cluster centroids are then calculated.

Another approach to partitional clustering is used in the Scatter/Gather system. Scatter/Gather uses two linear-time partitional algorithms, Buckshot and Fractionation are used for the refinement of the clusters and also apply HAC to select the initial cluster centers. The idea is to use these algorithms to find the initial cluster centers and then find the clusters using the assign-to nearest approach.

The single pass method, which is another method for partitional clustering, assigns each document to the cluster in which its most comparable representative is greater than a threshold. There is no iteration involved; the clusters are produced after a single scan of the data. As a result, the clustering is influenced by the sequence in which the documents are processed. These algorithms' simplicity and low computing complexity are their main merits.

Hierarchical Clustering Algorithm

Hierarchical Methods result in a tree-like representation. The clusters of documents highly similar to each other are nested within larger clusters of less similar documents. They can be agglomerative (building the tree from individual documents) or divisive (starting with the whole set and dividing it in to clusters).

(1) Determine all inter document similarities

(2) Form a cluster from the two closest documents or clusters

(3) Redefine the similarities between the new cluster and all other documents or clusters, leaving all other similarities unchanged. This step depends on the specific method
(4) Repeat steps 2 and 3 until all documents are in one cluster.
Some HACM's are

1) Single Link

- The similarity between two clusters is the maximum of the similarities between all pairs of documents such that one document is in one cluster and the other document is in the other cluster
- Each cluster member will be more similar to at least one member in that same cluster than to any member of another cluster

2) Complete Link

- Similarity between two clusters: minimum of the similarities between all pairs of documents
- Each cluster member is more similar to the most dissimilar member of that cluster than to the most dissimilar member of any other cluster (that is, more cohesive clusters than Single Link)

3) Group Average Link

- Similarity between two clusters: mean of the similarities between all pairs of documents, such that one document of the pair is in one cluster and the other document in the other cluster.

4.3 Difference Between Hierarchical and Partitional Clustering

The fact that each cluster begins as a separate cluster or singleton is the key distinction between hierarchical and partitional clustering, in conclusion. The nearest clusters are combined after each iteration. Up until there is just one cluster left, this process is repeated.
The Two-Step clustering method is an illustration of Hierarchical clustering.
In contrast, partitional clustering groups items that are close to the defined K number of clusters before performing the algorithm. The cluster's distance changes with each repetition. This process continues until either the halting requirement is satisfied or there is no longer any movement in the centroid of each cluster.
An example of Partitional clustering is the K-Means clustering method.
Typically, partitional clustering is faster than hierarchical clustering. Hierarchical clustering requires only a similarity measure, while partitional clustering requires stronger assumptions such as number of clusters and the initial centers.

5. Literature Survey

1) A systematic framework to discover pattern for web spam classification (2017)

Although many researchers have presented the different approach for classification and web spam detection still it is an open issue in computer science. Analysing and evaluating these websites can be an effective step for discovering and categorizing the features of these websites. There are several methods and algorithms for detecting those websites, such as decision tree algorithm. This paper, presents a systematic framework based on CHAID algorithm and a modified string-matching algorithm (KMP) for extract features and analysis of these websites.

2) Optimized Technique for Ranking Webpage on Search Engine Optimization (2018)

Search engine optimization is method that refers to the course of improving the traffic to a certain website by growing visibility of the location in the search engine results. SEO can be characterized as methodology used to elevate site keeping in mind the end goal to have a high rank that is, top outcome.

3) A Document Clustering Algorithm for Web Search Engine Retrieval System (2020)

As the number of available Web pages grows, it is become more difficult for users finding documents relevant to their interests. Clustering is the classification of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often proximity according to some defined distance measure. It can enable users to find the relevant documents more easily and also help users to form an understanding of the different facets of the query that have been provided for web search engine. A popular technique for clustering is based on K-means such that the data is partitioned into K clusters. In this method, the groups are identified by a set of points that are called the cluster centers. The data points belong to the cluster whose center is closest.

4) A Clustering-Based Approach for Assisting Semantic Web Service Retrieval (2018)

The discovery of suitable web services for a given task from a brokerage system is the core part of current Service Oriented Architecture (SOA). With the number of registered web services growing, organization of search results is critical for improving the utility of any service search engine. A clustering view of search results is more effective than traditional ranked-list style in helping users to navigate into relevant services quickly and accurately. In this paper, we will propose an efficient clustering algorithm for organizing returned services. The experiments validated the efficiency of the proposed method.

5) Search Engine Optimization Using Unsupervised Learning (2019)

Web has emerged as the most demanding tool for retrieving information over a large repository. As the amount of information on the world wide web grows, it becomes increasingly difficult to accurately find what we want. The existing search engines mostly display the content based on many factors and not just the quality of the content.

After crawling through the web and retrieving the information, they used ‘term frequency – inverse document frequency’ as our weighting algorithm, followed by ‘singular value decomposition’, for decomposition of the weighted matrix. Lastly, they used ‘spherical K-means’ and custom ranking algorithm to display rich content. In order to give more efficient results, our project presents a new algorithm to rank web pages in accordance to the relevance of the user’s query.

6) A novel approach for finding optimal search results from web database using hybrid clustering algorithm (2017)

The Internet provides an excellent extent of useful information that is sometimes arranged for its users, that makes it difficult to extract relevant information from various sources. So that, this paper proposes a hybrid Artificial Bee Colony and Improved K-means bunch algorithmic program provides all types data of data repository and has been terribly successful in dispersive information to users. For the encoded information units to be machine method intelligent, that is crucial for several applications like deep internet information assortment and net comparison searching, they have to be extracted out and allot substantive labels. This paper deals with the automated annotation of Search result records from the multiple internet databases.

7) Self-paced Learning for *K*-means Clustering Algorithm (2020)

The traditional K-means clustering algorithm is easily affected by the noise, outliers and falling into local optimal solution. This paper proposes a K-means clustering algorithm based on self-paced learning. Firstly, a best training subset is selected to construct the initial cluster model base on self-paced learning theory, and then enhances the generalization ability of the initial clustering model by adding sub-good subsets of samples one by one until the model performance is optimal or all training samples are used up.

8) Semantic Core Building of a Site Based on Clustering Algorithms

Websites are the main sources of information in the Internet. The ability to increase the ranking of the site for the search engine is particularly relevant in these circumstances. Engaging users with a demonstrating of a landing page on a search engine is the goal of search engine optimization. One of the steps of such optimization is a semantic core building. Semantic core is the most complete list of keywords that describes the theme and orientation of the site and matches its content. This paper presents semantic core building techniques for a site based on clustering algorithms.

The Base Paper Considered:
A Document Clustering Algorithm for Web Search Engine Retrieval System
This paper, proposed about the K-Means algorithm. Along with the comparison to all the partitional and hierarchical clustering algorithms.

6. Base Paper

A Document Clustering Algorithm for Web Search Engine Retrieval System

Document clustering was proposed mainly as a method of improving the effectiveness of document ranking following the hypothesis that closely associated documents will match the same requests. It is generally considered to be a centralized process. Examples of document clustering include web document clustering for search users.

- The main idea of clustering analysis method is to divide the original data set into different classes according to a certain metric so that the samples in the same category are as similar as possible, and the diversity between samples in different categories as large as possible.
- This paper gives an idea about document clustering, Web Page document clustering and clustering engines.
- The text-based web document clustering approaches characterize each document according to its content, that is, the words (or sometimes phrases) contained in it.

- Document clustering was proposed mainly as a method of improving the effectiveness of document ranking following the hypothesis that closely associated documents will match the same requests

Types of clustering algorithms proposed:

Partitional Clustering Algorithms

- The most common partitional clustering algorithm is k-means, which relies on the idea that the center of the cluster, called centroid, can be a good representation of the cluster. The algorithm starts by selecting k cluster centroids.
- Another approach to partitional clustering is used in the Scatter/Gather system. Scatter/Gather uses two linear-time partitional algorithms
- Buckshot and Fractionation are used for the refinement of the clusters and also apply HAC to select the initial cluster centers.
- The single pass method is another approach to partitional clustering which is based on the assignment of each document to the cluster with the most similar representative is above a threshold

Hierarchical Clustering Algorithms

- Single Link , the similarity between two clusters is the maximum of the similarities between all pairs of documents such that one document is in one cluster and the other document is in the other cluster
- Complete Link, similarity between two clusters: minimum of the similarities between all pairs of documents
- Group Average Link, similarity between two clusters: mean of the similarities between all pairs of documents, such that one document of the pair is in one cluster and the other document in the other cluster.

7. Overview of Clustering Engines

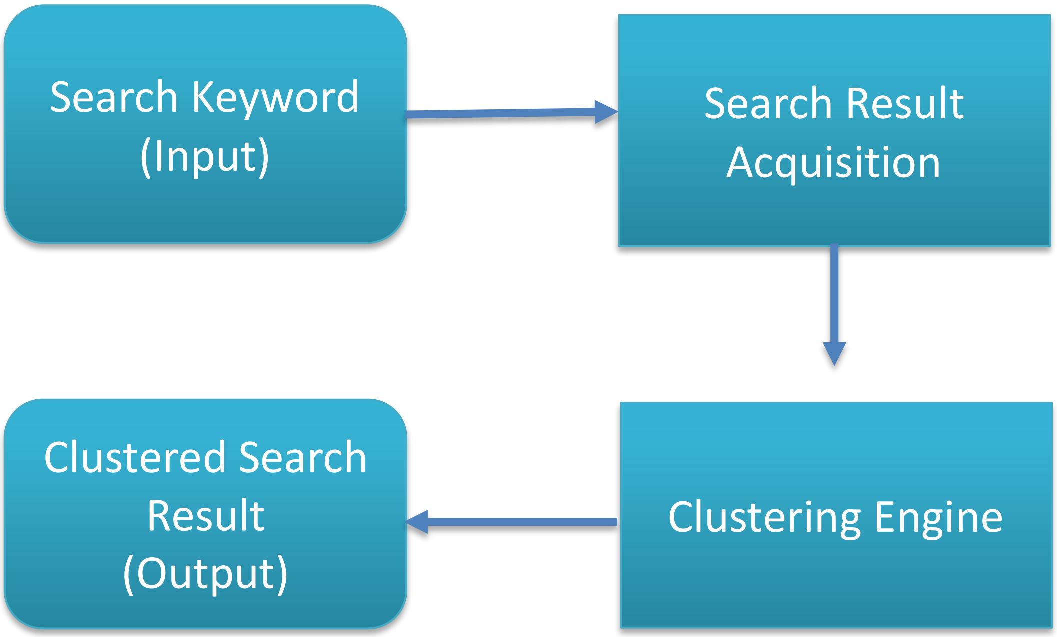


Figure 7.1 Overview of Clustering Engines

The output of the Web clustering engines ensures fast subtopic retrieval, quick topic exploration within unknown topics, and easy identification of relevant search results within the broad topic.

Main advantages of the cluster hierarchy is that:

o It allows for shortcuts to objects with similar meanings. The user can find related pages very easily since web clustering engines aggregate search results with the same meaning into the same cluster. Thus, the search period will be shorter.

o It enables greater subject comprehension. Web clustering engines are helpful for informational searches in uncharted or dynamic domains since they provide a high-level picture of the query.

o It encourages methodical investigation of search results. The user can evaluate hundreds of potentially relevant results without having to download and browse to further pages thanks to a clustering engine that condenses the material of numerous search results into a single view on the first result page.

8. Methodology Proposed

K-Means Algorithm

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.

It is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here 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.

> “It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.”

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the un-labeled dataset on its own without the need for any training.

Each cluster has a centroid assigned to it because the algorithm is centroid-based. This algorithm's primary goal is to reduce the total distances between each data point and its corresponding clusters.

The algorithm starts with an unlabeled dataset as its input, separates it into k clusters, and then continues the procedure until it runs out of clusters to use. In this algorithm, the value of k should be predetermined.

The k-means clustering algorithm mainly performs two tasks:

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

The below diagram explains the working of the K-means Clustering Algorithm:

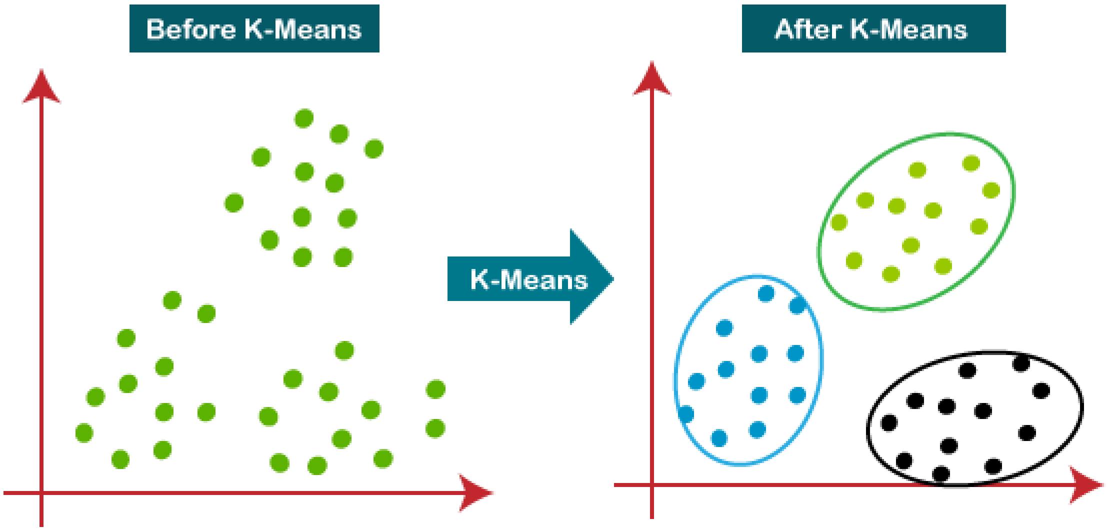


Figure 8.1 (K-Means Algorithm)

8.1 How does the K-Means Algorithm Works:

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

Step-7: The model is ready.

Illustrating the K-Means Algorithm with a example:

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

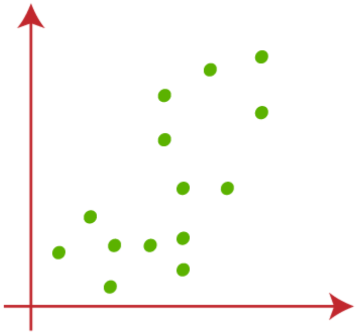


Figure 8.1.1

- Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image:

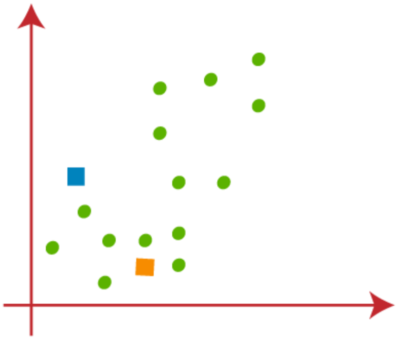


Figure 8.1.2

- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image:

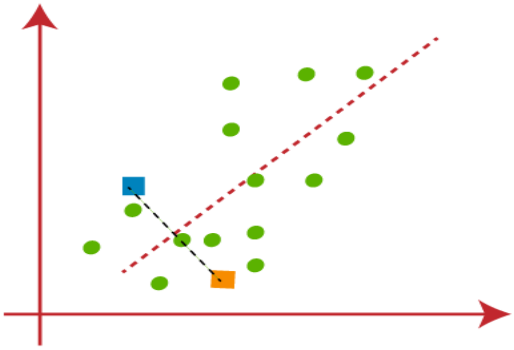


Figure 8.1.3

- From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.

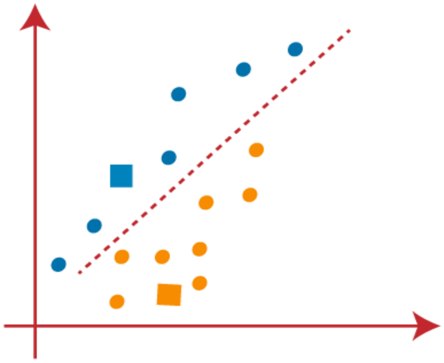


Figure 8.1.4

- As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:

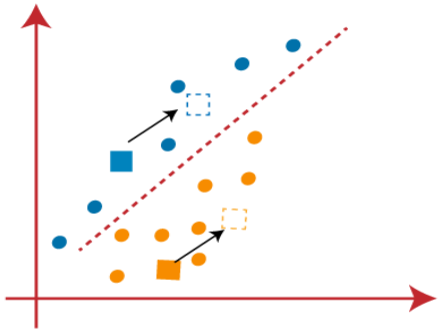


Figure 8.1.5

- From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

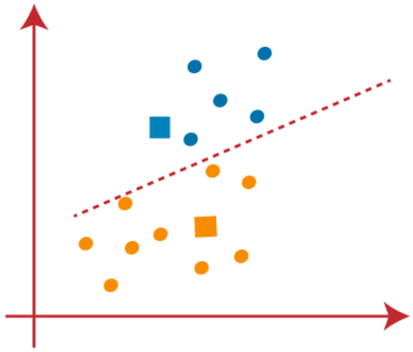


Figure 8.1.6

As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:

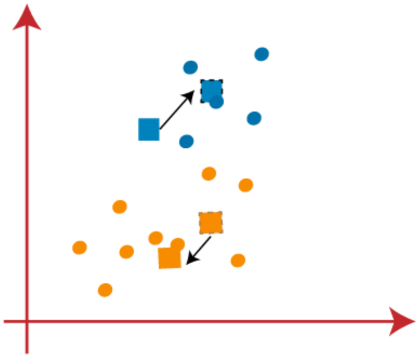


Figure 8.1.7

- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:

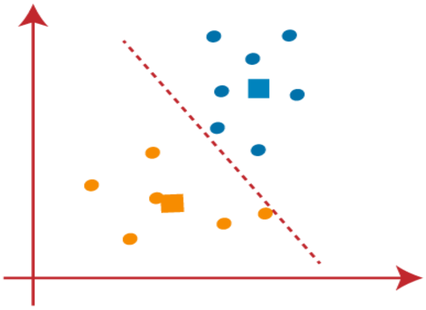


Figure 8.1.8

- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:

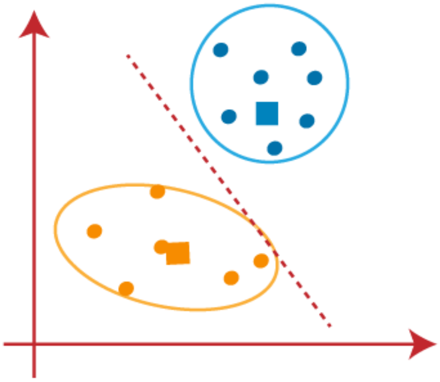


Figure 8.1.9

As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:

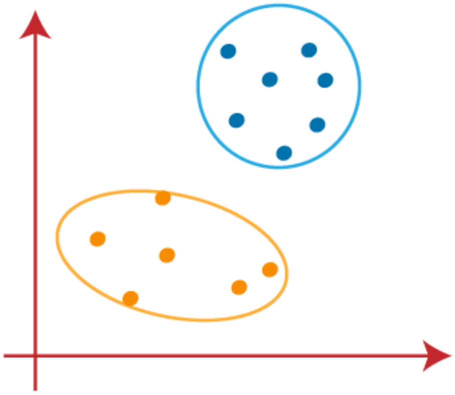
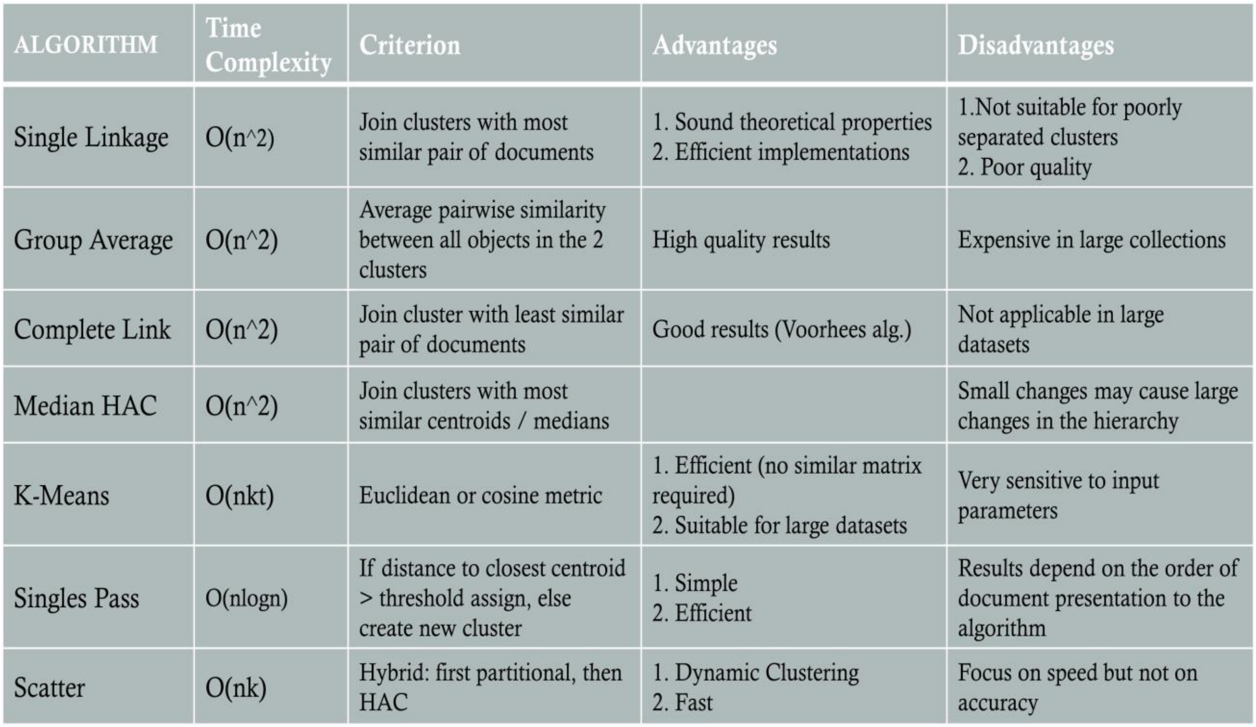


Figure 8.1.10

Thus, while using K-Means algorithm:

- It is suggested to normalize the data while dealing with clustering algorithms such as K-Means since such algorithms employ distance-based measurement to identify the similarity between data points.
- Because of the iterative nature of K-Means and the random initialization of centroids, K-Means may become stuck in a local optimum and fail to converge to the global optimum. As a result, it is advised to employ distinct centroids' initializations.

9. Comparison of Clustering Algorithms



ALGORITHM	Time Complexity	Criterion	Advantages	Disadvantages
Single Linkage	O(n^2)	Join clusters with most similar pair of documents	1. Sound theoretical properties 2. Efficient implementations	1.Not suitable for poorly separated clusters 2. Poor quality
Group Average	O(n^2)	Average pairwise similarity between all objects in the 2 clusters	High quality results	Expensive in large collections
Complete Link	O(n^2)	Join cluster with least similar pair of documents	Good results (Voorhees alg.)	Not applicable in large datasets
Median HAC	O(n^2)	Join clusters with most similar centroids / medians		Small changes may cause large changes in the hierarchy
K-Means	O(nkt)	Euclidean or cosine metric	1. Efficient (no similar matrix required) 2. Suitable for large datasets	Very sensitive to input parameters
Singles Pass	O(nlogn)	If distance to closest centroid > threshold assign, else create new cluster	1. Simple 2. Efficient	Results depend on the order of document presentation to the algorithm
Scatter	O(nk)	Hybrid: first partitional, then HAC	1. Dynamic Clustering 2. Fast	Focus on speed but not on accuracy

Table 1 (Comparison of Clustering Algorithms)

Table 1. compares several clustering algorithms in this example. The majority of them focus on the partitional and HAC algorithms, which are two of the most popular methods for text-based clustering.
As was already established, the single link approach among the HAC methods has the least complexity but produces the worst outcomes, while the group average produces the best. The general result is that the partitional algorithms are less difficult than the HAC when compared to partitional approaches.
Indeed, the complexity of the partitional algorithms is linear to the number of documents in the collection, whereas the HAC take at least O(n2) time.

Hierarchical algorithms are more efficient in handling noise and outliers. Another advantage of the HAC algorithms is the tree-like structure, which allows the examination of different abstraction levels. When k-means is run more than one times it may give better clusters than the HAC.

Finally, a disadvantage of the HAC algorithms, compared to partitional, is that they cannot correct the mistakes in the merges. A few words about the complexity and limitations of STC are the base cluster phase requires a time linear in the size of the documents, but hidden constants are high.

10. Search Engines

The Information Retrieval model component includes the Search Engine component. Its primary duty is to compare documents based on their document models by calculating the similarities between the documents. Clustering of results, which replaces document ranking in modern search engines, is the next level up.

A search engine has four components:

- Document processor indexes new documents. Indices are a mapping between words and what documents they appear in. Most engines are spider-based, so a crawl of the web for new documents and the updating of the index is automated.
- Query processor inspects a user’s query and translates it into something internally meaningful.
- Matching function uses the above internally meaningful representation to extract documents from the index.
- Ranking scheme positions the more-relevant documents on top, using some relevance measure.

11.Comparison document and web clustering

The development of clustering algorithms enables the organising of web pages in search engine results pages as well as the clustering of documents in database management systems. These algorithms provide a number of clustering engines, which automatically group the web pages that the search engines return. Some, like metacrawler.com, are working as metasearch engines. Others, including Grouper and Retriever, etc., serve as the clustering approaches' research test beds.

Clustering Type	Input	Online	Cluster label	GUI	Overlap
Document	Documents	No	Centroid	No	Crisp Clusters
Web	Snippets	Yes	Natural Language	Yes	Fuzzy Clusters

Table 2 (Comparison of Document and Web clustering)

- Document clustering was proposed to improve the effectiveness of document ranking following the hypothesis that closely associated documents will match the same requests.
- Alternatively, the web clustering engines group the ranked results and gives the user the ability to choose the groups of interest in an interactive manner.

12. Performance Evaluation

Based on the knowledge and experience gained from the current study, the following aspects can be considered while evaluating a Web search engine.

- Composition of Web Indexes

Whenever a Web search request is issued, it is the web index generated by Web robots or spiders, not the web pages themselves, that has been used for retrieving information. Therefore, the composition of Web indexes affects the performance of a Web search engine.

- Search Capability

A competent Web search engine must include the fundamental search facilities that Internet users are familiar with, which include Boolean logic, phrase searching, truncation, and limiting facilities

- Retrieval Performance

Retrieval performance is traditionally evaluated on three parameters: precision, recall and response time.

- Output Option

This evaluation component should be examined from two perspectives. One is the number of output options a Web search engine offers, whereas the other deals with the actual content of the output.

- User Effort

User effort refers to documentation and interface in this study. Well-prepared documentation and a user-friendly interface play a notable role in users' selection of Web search engines.

13. Conclusion

In report, presents the most important scientific and technical aspects of Web search result clustering. We have discussed the issues that must be addressed to build a Web clustering engine and have reviewed and evaluated a number of existing algorithms and systems.

As a result, the problems that need to be solved in order to construct a Web clustering engine have been explained, and a variety of search engine methods, systems, and performance metrics have been analysed. Despite the fact that there has already been a lot of research done in the area of clustering web documents, it is obvious that there are certain unresolved problems that require further study.

To improve the search result clustering,

The quality of the cluster labels and the coherence of the cluster structure must first be improved.

Second, incrementality, as new pages are always being added to the web and web pages are constantly changing.

Third, algorithms that support overlapping clusters should take into account the fact that online pages frequently relate to multiple topics.

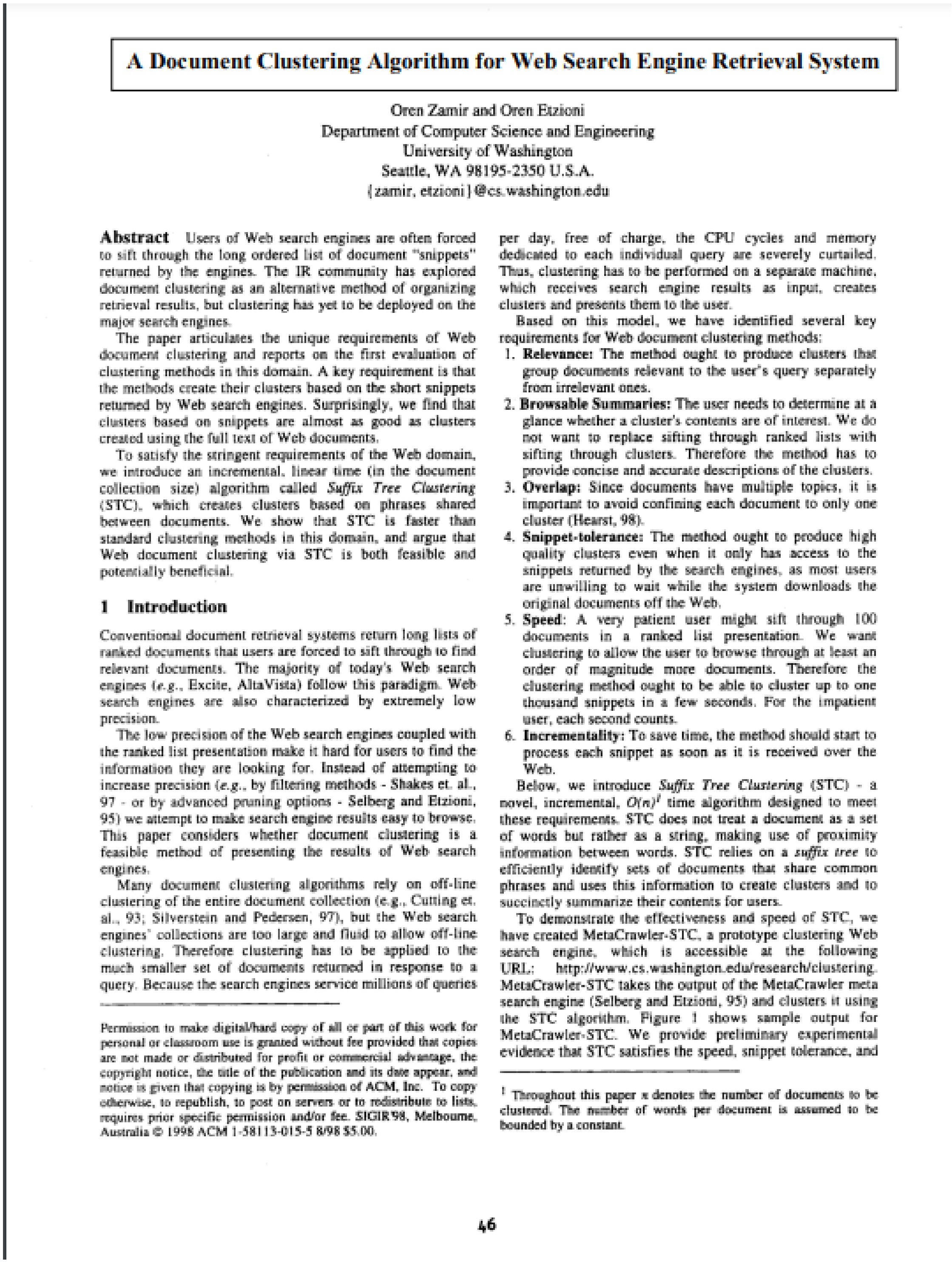
Fourth, consistency is still an issue. A cluster's contents may not always match its label, and searching via its subhierarchies may not always produce more precise results.

Fifth, more sophisticated visualization methods could be employed to give clearer overviews and direct engagement with grouped results..

14. References

1. “A systematic framework to discover pattern for web spam classification” 2017 8th IEEE Annual Information Technology, Date of Conference: 03-05 October 2017
2. “Optimized Technique for Ranking Webpage on Search Engine Optimization”, 2018 2nd International Conference on Micro-Electronics, Date of Conference: 20-21 September 2018
3. “A Document Clustering Algorithm for Web Search Engine Retrieval System”, Published in: 2010 International Conference, Date of Conference: 22-24 January 2019
4. “A Clustering-Based Approach for Assisting Semantic Web Service Retrieval”, Published in: 2008 IEEE International Conference on Web Services, Date of Conference: 23-26 September 2020
5. “Search Engine Optimization Using Unsupervised Learning”, Published in: 2019 5th International Conference on Computing, Date of Conference: 19-21 September 2019
6. “A novel approach for finding optimal search results from web database using hybrid clustering algorithm”, Published in: 2017 International Conference on Information Communication, Date of Conference: 23-24 February 2017
7. “Self-paced Learning for K-means Clustering Algorithm”, Published in: 2018
8. “Semantic Core Building of a Site Based on Clustering Algorithms”, Published in: 2020 10th International Conference on Advanced Computer, Date of Conference: 16-18 September 2020

15. Base Paper First Page



A Document Clustering Algorithm for Web Search Engine Retrieval System

Oren Zamir and Oren Etzioni
Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350 U.S.A.
{zamir, etzioni}@cs.washington.edu

Abstract Users of Web search engines are often forced to sift through the long ordered list of document "snippets" returned by the engines. The IR community has explored document clustering as an alternative method of organizing retrieval results, but clustering has yet to be deployed on the major search engines.

The paper articulates the unique requirements of Web document clustering and reports on the first evaluation of clustering methods in this domain. A key requirement is that the methods create their clusters based on the short snippets returned by Web search engines. Surprisingly, we find that clusters based on snippets are almost as good as clusters created using the full text of Web documents.

To satisfy the stringent requirements of the Web domain, we introduce an incremental, linear time (in the document collection size) algorithm called *Suffix Tree Clustering* (STC), which creates clusters based on phrases shared between documents. We show that STC is faster than standard clustering methods in this domain, and argue that Web document clustering via STC is both feasible and potentially beneficial.

1 Introduction

Conventional document retrieval systems return long lists of ranked documents that users are forced to sift through to find relevant documents. The majority of today's Web search engines (*e.g.*, Excite, AltaVista) follow this paradigm. Web search engines are also characterized by extremely low precision.

The low precision of the Web search engines coupled with the ranked list presentation make it hard for users to find the information they are looking for. Instead of attempting to increase precision (*e.g.*, by filtering methods - Shakes et. al., 97 - or by advanced pruning options - Selberg and Etzioni, 95) we attempt to make search engine results easy to browse. This paper considers whether document clustering is a feasible method of presenting the results of Web search engines.

Many document clustering algorithms rely on off-line clustering of the entire document collection (e.g., Cutting et. al., 93; Silverstein and Pedersen, 97), but the Web search engines' collections are too large and fluid to allow off-line clustering. Therefore clustering has to be applied to the much smaller set of documents returned in response to a query. Because the search engines service millions of queries per day, free of charge, the CPU cycles and memory dedicated to each individual query are severely curtailed. Thus, clustering has to be performed on a separate machine, which receives search engine results as input, creates clusters and presents them to the user.

Based on this model, we have identified several key requirements for Web document clustering methods:

1. **Relevance:** The method ought to produce clusters that group documents relevant to the user's query separately from irrelevant ones.
2. **Browsable Summaries:** The user needs to determine at a glance whether a cluster's contents are of interest. We do not want to replace sifting through ranked lists with sifting through clusters. Therefore the method has to provide concise and accurate descriptions of the clusters.
3. **Overlap:** Since documents have multiple topics, it is important to avoid confining each document to only one cluster (Hearst, 98).
4. **Snippet-tolerance:** The method ought to produce high quality clusters even when it only has access to the snippets returned by the search engines, as most users are unwilling to wait while the system downloads the original documents off the Web.
5. **Speed**: A very patient user might sift through 100 documents in a ranked list presentation. We want clustering to allow the user to browse through at least an order of magnitude more documents. Therefore the clustering method ought to be able to cluster up to one thousand snippets in a few seconds. For the impatient user, each second counts.
6. **Incrementality:** To save time, the method should start to process each snippet as soon as it is received over the Web.

Below, we introduce *Suffix Tree Clustering* (STC) - a novel, incremental, $O(n)$[1] time algorithm designed to meet these requirements. STC does not treat a document as a set of words but rather as a string, making use of proximity information between words. STC relies on a *suffix tree* to efficiently identify sets of documents that share common phrases and uses this information to create clusters and to succinctly summarize their contents for users.

To demonstrate the effectiveness and speed of STC, we have created MetaCrawler-STC, a prototype clustering Web search engine, which is accessible at the following URL: http://www.cs.washington.edu/research/clustering. MetaCrawler-STC takes the output of the MetaCrawler meta search engine (Selberg and Etzioni, 95) and clusters it using the STC algorithm. Figure 1 shows sample output for MetaCrawler-STC. We provide preliminary experimental evidence that STC satisfies the speed, snippet tolerance, and

[1] Throughout this paper *n* denotes the number of documents to be clustered. The number of words per document is assumed to be bounded by a constant.