Optimal Search of Centroid for Open Source Intelligence Purpose using Partitioning Algorithm

Mohd. Shajid Ansari
Asst.Prof.
Deptt. of CSE
RSR-RCET, Bhilai

Manuraj Jaiswal
Asst.Prof.
Deptof CSE
RSR-RCET, Bhilai

ABSTRACT
In internet billions amount of information published through no of ways like web pages, social media and website. Searching and analyzing these data is very complex task. Using partitioning algorithm for open source intelligence purpose optimal search can be implemented which help to convert unstructured data to structured data and also analysis and extraction the information significantly. In partitioning algorithm we use binary search technique. Each algorithm has its own advantages, limitations and shortcomings. Therefore, introducing novel and effective approaches for data clustering is an open and active research area. The binary search algorithm for data clustering that not only finds high quality clusters but also converges to the same solution in different runs.

Open Source Intelligence (OSINT) aims at presenting valuable information based on publicly available data. As it might be expected, the Internet is a primary example of such data source. By applying text mining tools on a myriad of available services: online news, blogs, mailing lists, forums, portals, and a great amount of insight might be provided into almost any topic.

Keywords
Text Mining, OSINT, K – Means Algorithm, Agglomerative Algorithm.

1. INTRODUCTION
In internet billions amount of information published through no of ways like web pages, social media and website. Searching and analyzing these data is very complex task. Using partitioning algorithm for open source intelligence purpose optimal search can be implemented which help to convert unstructured data to structured data and also analysis and extraction the information significantly. In partitioning algorithm we use binary search technique.

Each algorithm has its own advantages, limitations and shortcomings. Therefore, introducing novel and effective approaches for data clustering is an open and active research area. The binary search algorithm for data clustering that not only finds high quality clusters but also converges to the same solution in different runs.

Open Source Intelligence (OSINT) aims at presenting valuable information based on publicly available data. As it might be expected, the Internet is a primary (if not perfect) example of such data source. By applying text mining tools on a myriad of available services: online news, blogs, mailing lists, forums, portals, and a great amount of insight might be provided into almost any topic.

While originally associated with governments, OSINT is also an area of interest for companies (making research on the market and/or the competition) or even personals (analyzing some specific topic). Some of the typical use cases include:

- Collecting information on given topic (e.g. related to suspicious financial operations),
- Searching information in given context (e.g. what were the financial operations of XYZ in 2009?),
- Analyzing social networks, finding out connections between entities (e.g. A knows B and B knows C, which works for XYZ Inc.),
- Analyzing gathered information (e.g. trends on how many articles about XYZ were published in 2009 and how many in 2008?).
- OSINT creates a hard problem in a computer supporting aspect, and requires much more implementation effort than, for example, keyword search engines.
- OSINT tools are complicated itself, they cover both retrieving the information as well as searching, filtering, extracting and analysis,
- Amount of processing done by OSINT system is typically much larger than that done by the typical search engine; very often, a large amount of semantic processing is included.

2. Architecture of OSINT
Design Requirements:

- OSINT is actually a process, in which data must be first collected and then be a subject to filtering and extraction,
- system aims at being interactive, allowing users to retrieve required data in a timely manner,
- A significant number of use-cases (specific intelligence requirements) are supported. The OSINT system can be broken down into three separate aspects:

1. Collecting – i.e. where to get the data from?
2. Extracting and analyzing – i.e. what the data contain?
3. Presenting – i.e. what does it mean?
4.

Taking this into the realm of Internet–based data sources, a high-level architecture layout is proposed in fig. 1. The core of the presented system is split into three main functional parts:

1. Collecting Data

2. Preliminary Processing Chain
3. Request Processing Chain

Such design is chosen because after receiving the data from the crawlers, computationally intensive tasks are started for extracting data [2, 3, 4] required by the last phase, which can be seen as a stack of filters. It works for the user's sake, is controlled by him and allows for retrieval of both the documents and the knowledge derived.

Collecting Data

Instead of relying on a single crawling solution, the system provides a REST (Representational State Transfer) style [5] interface, which allows for simple communication with any kind of data collection subsystem. A number of such processes asynchronously send a list of retrieved documents. Each of the entries contains text, unique id and optional metadata fields, such as:

- author name
- publication date
- language or thread id

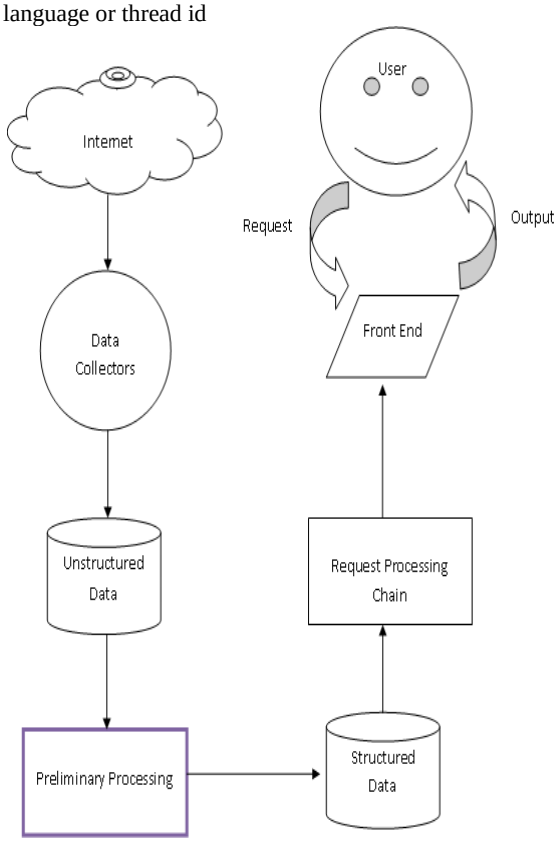


Figure-1

The currently available list of data collectors consists of:

- Web crawler – developed by PPBW, basing on Apache Droids5; the solution also supports a meta-search – in such case, a query is sent to a web-search engine and the results are used as a seed for the crawler,
- Forum data collector – supports phpBB, IPBoard and vBulletin,
- Blog data collector – uses DOM (Document Object Model) for extracting meaningful contents from engines such as BlogSpot, Wordpress, Blox.pl and more,
- Social networks data collector – supports Facebook and Twitter, via their API,
- Database collector – allows for easily retrieving data from relational databases, as well as from various document collections; the latter is often used for testing and research based on widely available text corpora.

There are two methods for collecting data:

1. Ad-hoc operation
2. Simple message queue

In ad – hoc operation data collected manually from selected target with proper condition and criteria.

In simple message queue provides list of starting addresses (a seed) with parameters. The data collectors periodically query the server and, if necessary, the acquisition is being started.

Some of the clustering algorithms:

➢ K-means clustering

K-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more

Although K-mean is a fast and simple algorithm but it is very sensitive to the selection of the initial centroids, in other words, the different

centroids may produce completely different results. Another drawback of K-means is that, it may trap in local optima solution.

➢ Agglomerative clustering

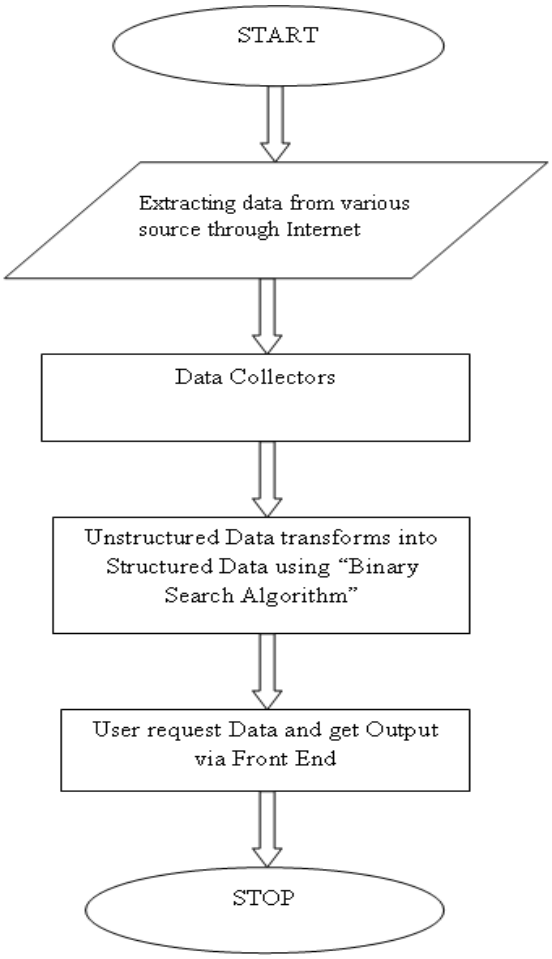
The algorithm forms clusters in a bottom-up manner, as follows:

1. Initially, put each article in its own cluster.
2. Among all current clusters, pick the two clusters with the smallest distance.
3. Replace these two clusters with a new cluster, formed by merging the two original ones.
4. Repeat the above two steps until there is only one remaining cluster in the pool.

Thus, the agglomerative clustering algorithm will result in a binary cluster tree with single article clusters as its leaf nodes and a root node containing all the articles.

3. PROPOSED APPROACH

Flow Chart of Proposed Method:



- **Binary Search Algorithm**

In this algorithm cluster will be a input and partitioned into k part, each partition will be treated as cluster. In this method we takes a initial centroids and assign data object to nearest centroids and find objective function values

The proposed algorithm is a partition-based algorithm that takes the number of the clusters as input parameter. The algorithm organizes the objects into exactly k partitions where each partition represents a cluster. The proposed method is a simple and straightforward method. This method generates initial centroids and assigns each data object in the dataset to the nearest centroids and evaluates the objective function. It then gradually moves to a new centroid and evaluates the objective function again and compares to the previous objective value. If the new centroid provides a lower objective value, then the new centroid is kept and movement continues along the same direction. If the objective value is higher, the new centroid is ignored and movement continues in the opposite direction. The move will be stopped when the maximum number of iterations is reached or the quality of solution is acceptable. We explain the generation and relocation of the centroids in details in the following.

The generation of the initial centroids is done in this proposed algorithm works as follows. First of all the algorithm establishes a set of k initial centroids. To choose the initial centroids from different parts of the test dataset and to keep them as far as possible, the input dataset are divided into k equal parts in a simple way:

$$G = (Max(dataset) - (Min(dataset)) / K$$

Where Max(dataset) and Min(dataset) are correspond with data objects that their attributes are the maximum and minimum values in whole of the test dataset respectively. k is the predefined number of clusters for the given dataset and G is the range of the produced partitions. After that, the initial centroids are generated in the following way:

$$C_i = Min(dataset) + (i - 1) \times G , i = 1 , 2 , 3 \ldots \ldots, k$$

At the second stage, the algorithm assigns each object in the dataset to exactly one centroid based on the distance measure. Each object is assigned to those centroid that is the nearest to the given object. In such a way a starting clustering is achieved. In the further steps the algorithm iteratively improves the quality of the clustering by relocation of the centroids.

The next step is that the algorithm changes the locations of the centroids by adding or subtracting a special value to the current value of their attributes. This is done to find the best location of

the centroid in the respective cluster. We will call this special value as the step size of movement (SSM). The initial value of the SSM for an attribute will be equal to the maximum value of that attribute in the test dataset and it will change during the search process. It also may be different for different attributes and different centroids. The relocation of the centroids will be done in the following way. For each centroid, algorithm selects the first attribute and adds its corresponding SSM to its current value. By doing this, the location of the respective centroid is changed. Then each data object is reassigned to the nearest centroid. In such a way there is a possibility of moving some data objects from one cluster to another cluster. So the algorithm evaluates the objective function again and compares to the previous objective value. If an improvement happened, then the new point is kept and movement continues along the same direction. Otherwise, the new point is ignored and movement proceeds in the opposite direction by changing the sign of its SSM. Subsequently, algorithm explores the new direction using the same scenario. If an improvement occurs in the objective function, algorithm keeps the new point, otherwise ignores it and divides the SSM by 2. This is done in order to search spaces close to the current centroid more precisely in next iterations. This process is followed for the other attributes in the current centroid and also for other centroids, respectively. After reaching the last attribute in the last centroid, this procedure restarts from the first attribute of the first centroid. The whole process iterates until a termination criterion is reached.

4. CONCLUSION

A binary search algorithm for data clustering is proposed in this work. A set of initial centroids is generated using the test dataset. The proposed algorithm thoroughly explores around of the initial centroids to find optimal locations for the clusters centroids. There are several advantages for the proposed algorithm: it has simple structure and it is easy to implement, it is reliable and precise and in all of runs produces the same results and finally the quality of the solutions found by the proposed algorithm is better than other test algorithms.

5. REFERENCES

[1] CLUO: WEB – SCALE TEXT MINING SYSTEM FOR OPEN SOURCE INTELLIGENCE PURPOSE COMPUTER SCIENCE 14 (1) 2013

[2] Cover T., Thomas J.: Elements of Information Theory. Wiley, 1991.

[3] Jurafsky D., Martin J. H.: Speech and Language Processing Prentice Hall, 2 ed. 2008.

[4] Manning C., Raghavan P., Sch¨utze H.: Introduction to Information Retrieval. Cambridge University Press, 1 ed., 2008.

[5] Fielding R. T.: Architectural styles and the design of network-based software architectures. PhD thesis, 2000.

[6] S.Z. Selim, K. Alsultan, A simulated annealing algorithm for the clustering problem, Pattern Recognition 24 (10) (1991) 1003–1008.

[7] K.S. Al-Sultan, A Tabu search approach to the clustering problem, Pattern Recognition 28 (9) (1995) 1443–1451.

[8] A.K.Qin,P.N.Suganthan, Kernel neural gas algorithms with application to cluster analysis, in: Proceedings—International Conference on Pattern Recognition, 2004.

[9] P.S. Shelokar, V.K. Jayaraman, B.D. Kulkarni, An ant colony approach for clustering, Analytica Chimica Acta 509 (2) (2004) 187–195.

[10] D. Karaboga, C. Ozturk, A novel clustering approach: artificial bee colony (ABC) algorithm, Applied Soft Computing 11 (1) (2011) 652–657.

[11] M. Fathian, B. Amiri, A. Maroosi, Application of honey-bee mating optimization algorithm on clustering, Applied Mathematics and Computation 190 (2) (2007) 1502–1513