International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2020 IJSRCSEIT | Volume 6 | Issue 2 | ISSN : 2456-3307
DOI : https://doi.org/10.32628/CSEIT206216

# Frequent Pattern Mining over Unstructured Data using Semi-Structured Doc-Model and Pattern Ranking

**Sudhir Tirumalasetty*[1], A. Divya[2], D. Rahitya Lakshmi[3], Ch. Durga Bhavani[4], D. Anusha[5]**

*[1]Department of Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh, India

[2, 3, 4, 5] Department of Computer Science & Engineering, Vasireddy Venkatadri Institute of Technology, Guntur, Andhra Pradesh, India

## ABSTRACT

Frequent pattern mining is an essential data-mining task, with a goal of discovering knowledge in the form of repeated patterns. Many efficient pattern-mining algorithms have been discovered in the last two decades, yet most do not scale to the type of data we are presented with today, the so-called “Big Data”. Scalable parallel algorithms hold the key to solving the problem in this context. This paper reviews recent advances in parallel frequent pattern mining, analysing them through the Big Data lens. Load balancing and work partitioning are the major challenges to be conquered. These challenges always invoke innovative methods to do, as Big Data evolves with no limits. The biggest challenge than before is conquering unstructured data for finding frequent patterns. To accomplish this Semi Structured Doc-Model and ranking of patterns are used.

**Keywords :** Data Mining, Doc-Model, Frequent Pattern Mining, Pattern Rank, Unstructured Data

## I. INTRODUCTION

### A.Traditional File System

The customary recording framework (TFS) is a technique for putting away and masterminding PC documents and the data in the record (information). Fundamentally, it sorts out these documents into a database for the capacity, association, control, and recovery by the PC's working system [1].

Record based frameworks were an early endeavor to modernize the manual documenting framework. Record based framework is an assortment of utilization programs that perform administrations for the end-clients, for example, refreshing, addition, and erasure adding new documents to database and so on. Each program characterizes and deals with its information.

At the point when a PC client needs to store information electronically, they should do as such by setting information in records. Records are put away in explicit areas on the hard plate (catalogs). The client can make new documents to put information in, erase a record that contains information, rename the document, and so on which are known as document the executives; a capacity gave by the Operating System (OS).

The improvement of the record based framework lead to database the board framework as a result of the burdens of conventional document framework, for example, information repetition, information

irregularity, trouble in getting to information, constrained information sharing, uprightness issues, atomicity issues, simultaneous access abnormalities, and security issues.

### B. Relational Database Systems

Because of a great deal of wasteful aspects of the conventional record framework during the 1970s line thought of the social hypothesis that prompted the improvement of the Relational Database Management Systems (RDBMS) as an answer for the difficulties presented by the level document database framework in the prior years. Capacity of information in RDBMS was finished utilizing Tables. Standard fields and records are spoken to as sections (fields) and columns (records) in a table. Their significant preferred position was the capacity to relate and record data. Security was upgraded in RDBMS and they were additionally ready to adjust to impressive development of information. Organized Query Language, SQL is the programming language utilized for questioning and refreshing social databases. For quite a while RDBMS has been the favored procedure for information the executives purposes. Be that as it may, RDBMS powerlessness to deal with current remaining tasks at hand has offered ascend to adaptability, execution and accessibility issues with its unbending diagram design [2].

### C. Big-Data

What is large information? Up until now, there is no all-around acknowledged definition. In Wikipedia, huge information is characterized as "a widely inclusive term for any assortment of informational indexes so enormous and complex that it gets hard to process utilizing conventional information preparing applications" [3]. Contrasted with customary information, the highlights of huge information can be described by 5V, to be specific, colossal Volume, high Velocity, high Variety, low Veracity, and high Value. The primary trouble in adapting to huge information does not just lie in its immense volume, as we may reduce somewhat this issue by sensibly growing or broadening our figuring frameworks. Overall, the genuine difficulties revolve around the differentiated information types (Variety), opportune reaction prerequisites (Velocity), and vulnerabilities in the information (Veracity). Because of the expanded information types, an application frequently needs to manage conventional organized information, yet additionally semi-organized or unstructured information (counting content, pictures, video, and voice). Opportune reactions are likewise testing in light of the fact that there may not be sufficient assets to gather, store, and procedure the large information inside a sensible measure of time. Finally, recognizing genuine and bogus or solid and untrustworthy information is particularly testing, in any event, for the best information cleaning techniques to wipe out some natural eccentrics of information.

*Volume:* Organizations gather information from an assortment of sources, including business exchanges, keen (IoT) gadgets, mechanical gear, recordings, web based life and the sky is the limit from there. Before, putting away it would have been an issue – yet less expensive stockpiling on stages like information lakes and Hadoop have facilitated the weight.

*Speed:* With the development in the Internet of Things, information streams in to organizations at an uncommon speed and should be taken care of in an opportune way. RFID labels, sensors and savvy meters are driving the need to manage these deluges of information in close constant.

*Assortment:* Data comes in a wide range of arrangements – from organized, numeric information in customary databases to unstructured content reports, messages, recordings, sounds, stock ticker information and monetary exchanges.

*Inconstancy:* notwithstanding the expanding speeds and assortments of information, information streams are flighty – changing frequently and fluctuating significantly. It's difficult; however organizations need to realize when something is inclining in internet based life, and how to oversee day by day, regular and occasion activated pinnacle information loads.

*Veracity:* Veracity alludes to the nature of information. Since information originates from such a significant number of various sources, it's hard to connect, coordinate, wash down and change information across frameworks. Organizations need to interface and connect connections, pecking orders and various information linkages. Something else, their information can rapidly winding out of control [4].

### D. Doc-based

In any field tremendous measure of informational collections are being utilized where social database models are not adequate to keep up all the datasets so report based models are presented. A record situated database, or archive store, is a PC program intended for putting away, recovering and overseeing report arranged data, otherwise called semi-organized data [5].

Archive arranged databases are one of the fundamental classifications of NoSQL databases, and the fame of the expression "record situated database" has developed with the utilization of the term NoSQL itself. XML databases are a subclass of record arranged databases that are upgraded to work with XML reports. Diagram databases are comparative, however include another layer, the relationship, which permits them to connect records for quick traversal.

Archive situated databases are naturally a subclass of the key-esteem store, another NoSQL database idea. The distinction lies in the manner the information are prepared; in a key-esteem store, the information are viewed as characteristically murky to the database, while a record arranged framework depends on inside structure in the report so as to remove metadata that the database motor uses for additional enhancement. In spite of the fact that the thing that matters is regularly debatable because of apparatuses in the frameworks, reasonably the record store is intended to offer a more extravagant involvement in present day programming strategies.

Archive databases balance unequivocally with the conventional social database (RDB). Social databases for the most part store information in independent tables that are characterized by the software engineer, and a solitary item might be spread over a few tables. Archive databases store all data for a given item in a solitary occurrence in the database, and each put away article can be not the same as each other. This takes out the requirement for object-social mapping while at the same time stacking information into the database.

There are different highlights which makes archive databases not the same as social databases like Intuitive Data Model(Faster and Easier for Developers), Flexible Schema(Dynamically adjust to change), Universal(JSON Documents are Everywhere), Powerful(Query Data Anyway You Need), Distributed(Resilient and Globally Scalable).

The following sections include literature, followed by the implementation of proposed model with findings.

## II. LITERATURE

### A.Why NoSQL

A document orient database or a NoSQL record store is a cutting edge approach to store information in JSON design instead of straightforward lines and sections. It permits you to communicate information in its characteristic structure the manner in which it's intended to be. For as far back as 40 years, social databases have commanded the database business.

Social databases arrange information in tables of lines and segments and create connections between them. These connections are viewed as the consistent associations between the tables and are characterized based on the information itself.

Anyway, social databases could not adapt up to the difficulties expressed beneath.

1. Agile Software Development requires versatile procedures and faster shipment of programming. This implies the hidden database should empower this strategy. Conversely, lines and sections are intrinsically inflexible information structure. One section expansion affects the entire table.

2. With Relational databases, an adjustment in the information model methods an engineer demands changes from a database executive. This human escalated procedure can take unreasonably yearn for developing and developing organizations that depend on rapid arrangement.

3. Ultimately, information communicated in lines and sections is an unnatural method to store data. The explanation being, a line in a RDBMS is only a level information structure, with information separated into sections, though, information patterns are considerably more perplexing along these lines request more flexibility [7].

### B. Various NOSQL Models:

There are various models of NoSQL databases. Here is the list of NoSQL databases

1. **Key-Value databases**(It has a big hash table of keys and values)
2. **Document based databases**(It stores documents made up of tagged elements)
3. **Column oriented databases**(Each storage block contains data from only one column)
4. **Graph databases**(A network database that uses edges and nodes to represent and store data) [6]

### C. Why MongoDB:

MongoDB is one of the report based capacity model of NoSQL databases. In MongoDB, records [8] that utilization a B-tree information structure are created at assortment (proportional to a table in connection databases) level. To help an assortment of information and inquiries, there are various kinds of lists accessible:

1) Default _id - this exists as a matter of course on the _id field of assortments and furthermore fills in as a one of a kind id;

2) Single Field - this is a client characterized list on a solitary field of a record;

3) Compound Index - this is a client characterized list on different fields;

4) Multi key Index - this is utilized to file a field that holds a variety of qualities;

5) Geospatial Index - this is an extraordinary list that MongoDB (in contrast to different models) gives to help geospatial questions and has two sorts: (I) 2d spatial file for planer geometry inquiries, (ii) 2sphere spatial list for round geometry inquiries;

6) Text Index - this is the file type that enables scan for the string or content to content in an assortment; and

7) Hashed Index-this helps list the hash of the estimations of a field.

## III.PROPOSED MODEL

### A. Why:

Big Data is being used in many fields, and it contains many number of datasets where it is not possible to find all the frequent patterns in the various datasets. This proposed model helps us to take the required content from those large datasets and can be differentiated based on our usage of the data.

### B. Model Explanation:

In this model we preferred a website (www.vvitguntur.com) where we extracted the content of all the staff data based on some particular tags in the inspect page of the website.

A database with a name departments as shown in Figure 2 is created and various collections are created in the database with various department names here,

we prefer to take csedepartment as a collection name as shown in Figure 3.

From the website based on the name, education, profession and reference id each staff member data is created as a document sample in the collection as shown in Figure 4..

## IV. IMPLEMENTATION

For implementing this proposed model, we used Spyder (Python 3.7) for implementing the code. The below implemented code helps us to get the content from the website and store it in the Mongo dB database, where each staff field is stored as a document in the collection.

The below code is used for extracting data from a website and stores the data in the MongoDB server. When we run this program, the program shows the ObjectId of each document in each collection in the database as shown in Figure 1.

### A. Code Snippet:

*Code for extracting Data from website*

```
html=requests.get("https://vvitguntur.com/cse-faculty")
        res=BeautifulSoup(html.content,'html.parser')
        ta=res.find("section",{"id":"s5_below_body_wrap"})
        tags=ta.find_all("a")
```

*Code for storing data in MongoDB server*

```
myclient= pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["departmentss"]
mycol = mydb["csedepartment"]
dict={"name":name,"education":education,"profession":profession,"ref":("https://www.vvitguntur.com"+lists[i]),"information":info}
   data.append(dict)
x = mycol.insert_many(data)
print(x.inserted_ids)
```


**Figure 1**: Shows the ObjectId of the each document where the data is stored in MongoDB server

After executing this code, it shows some object ids in the console, where the ids are references of each staff member.

In the implementation of this code, if the database is not present in the databases list in MongoDB, then it creates the database with the given name and create collections with an object id for each document in the collection. The result of this code is as follows.

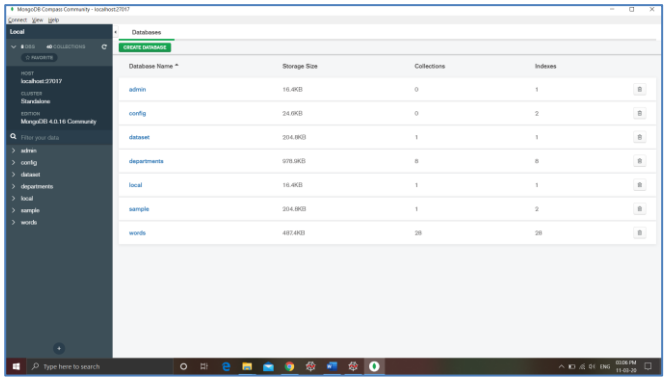

**Figure 2:** Database in MongoDB server

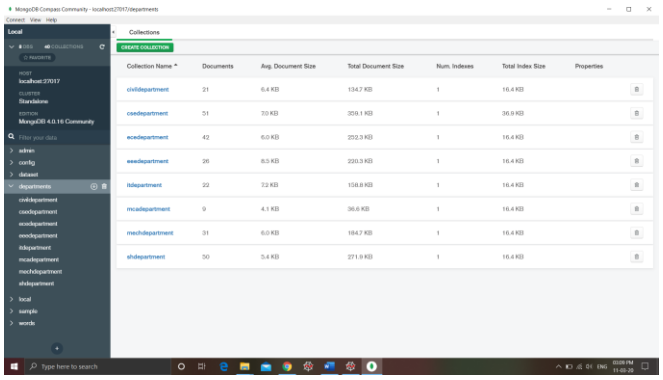

**Figure 3:** Collections in the database after executing the code

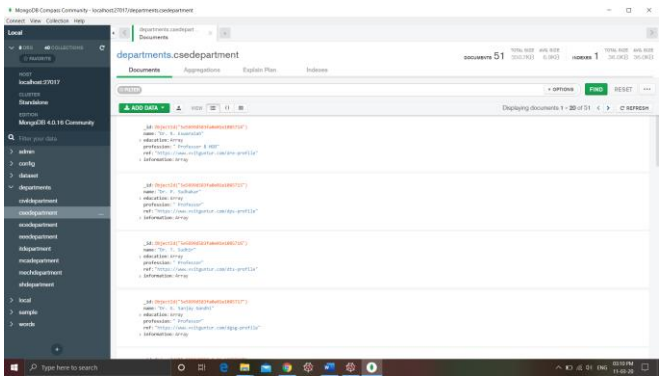

**Figure 4:** Each collection containing the data of each staff member

As our aim is to produce frequent patterns from the huge datasets here is another sample code where when it is implemented. This shows all the frequent items related to the researches, seminars, journals implemented by lectures. The proposed model asks the user to enter the word he need to search for. If the word the user has given is found in the database it retrieves the data from the database and shows the word count to the user as per the given word. If the word given by the user is not present in the database it displays a message that, "The word you are trying to search is not found in the database. Please try for another word."

This code checks each and every document for the user given word. In this program it stores the data retrieved by the user given word in the database with the word given by user as the collection name. If the user is trying to check for the word, he previously checked then it retrieves data from the database without storing the data again in the database. If the word user is trying to search is not in his previously searched list then it stores the data in database and retrieve the data in the descending order based on the count of the words in each document in each collection as shown in Figure 5.

### B. Code Snippet:

*Extract user needed data from the server*

```
for x in mycol.find({"information":{"$regex":words}}):
        for k in range(len(x['information'])):
            linklist.append(x['information'][k])
        counts=0
        for m in linklist:
            if words in m:
                counts+=1

dict={"name":x['name'],"department":j,"count":counts
}
```

*Code to store the extracted data in the server*

```
mycol=mydb1[words]
        el=mycol.insert_many(data)
        mydac=mycol.find().sort("count",-1)
        for h in mydac:
            print(number,".                ",h['name'],"--
>",h['department'],"-->",h['count'])
```

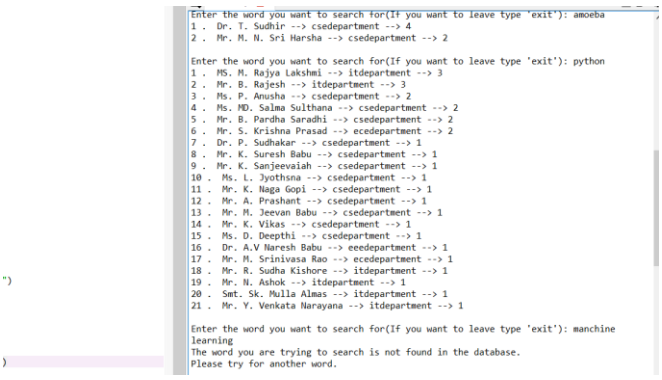


**Figure 5:** Retrieving data based on user needs

## V. ADVANTAGES

- 
- Large volume of data can be stored in MongoDB
- Extract data from any website and compare the data in them.
- Less time, more efficiency
- Retrieval is easy without searching for the data in various files.
- Easy access over the data
- Less complexity

## VI.CONCLUSION

We presented a reliable general method for ranking frequent patterns with respect to the data. This helps one to understand and retrieve the data easily from the website. This helps the user to access the data as the unstructured data is converted to semi-structured data. The unstructured data consists of irregularities and ambiguities due to which user faces problems in retrieving the data so the data is converted to semi-structured data which helps in removing some of the irregularities and ambiguities. This helps in pharmaceutical industries, educational institutes, finance and marketing, etc used to find the required data based on their need. This project can be used and modified based on the user needs. The data can be taken in the form of text, images, XML, HTML and modified based on the requirements.

## VII. REFERENCES

[1] UKEssays https://www.ukessays.com/essays/information-technology/traditional-file-systems-and-database-management-information-technology-essay.php

[2] Innocent Mapanga, Prudence Kadebu "Database Management Systems: A NoSQL Analysis", International Journal of Modern Communication Technologies & Research (IJMCTR) ISSN: 2321-0850, Volume-1, Issue-7, September 2013

[3] Big Data, https://en.wikipedia.org/wiki/Big_data

[4] SAS Insights "History of Big Data", https://www.sas.com/en_in/insights/big-data/what-is-big-data.html

[5] Document-oriented database, https://en.wikipedia.org/wiki/Document-oriented_database

[6] Rupali Arora, Rinkle Rani Aggarwal "Modeling and Querying Data in MongoDB", International Journal of Scientific & Engineering Research, Volume 4, Issue 7, July-2013,141 ISSN 2229-5518

[7] NCache, https://www.alachisoft.com/nosdb/document-databases.html

[8] Bertino, E., Beng, C. O., Ron, S.D., Kian, L.T., Justin, Z., Boris, S., & Daniele, A. (2012) Indexing techniques for advanced database systems. Springer Publishing Company, Incorporated.

**Cite this article as :**

Sudhir Tirumalasetty, A. Divya, D. Rahitya Lakshmi, Ch. Durga Bhavani, D. Anusha, "Frequent Pattern Mining over Unstructured Data using Semi-Structured Doc-Model and Pattern Ranking ", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 6 Issue 2, pp. 36-42, March-April 2020. Available at doi : https://doi.org/10.32628/CSEIT206216
Journal URL : http://ijsrcseit.com/CSEIT206216