Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 1, January 2021: 404-412

Big Data Analytics Approach for Structured and Unstructured Data

Rahul Kumar[1], Subodh Kumar[2] and Javed Wasim[1]
[1]Department of Computer Engineering &Applications, Mangalayatan University Aligarh, U.P., India
[2]Velocis Systems Private Limited, Noida
*Ema*il: rahulbca18@gmail.com[1]

ABSTRACT: The volume of data in the world is growing very fast and generated from verity of sources like social media, sensors airline industry or scientific data in different formats. Biggest challenge is how to infer meaningful insights from such a variety and big data along with concern of data storage and management of fast-growing data. The size of the databases used in today's enterprises has been growing at exponential rates day by day. Hence, retrieval and extraction of the information is essential works and importance in semantic web areas to fulfil the industries requirement to quickly process and analyseexponential growing the big data. Data torrential from various sources may be structured or unstructured in nature. Structured data refers to a relatively well-organized information but high in volume, which can also analyses by big data tools in better way. As Traditional RDBMS are not very efficient to queries on textual filter condition, In contrast to structured data, unstructured data can be considered as information, which does not, comes in a pre-defined data format, well organized data storage model, or cannot be handle by legacy RDBMS models. It is assumed to be fastest growing type of data, e.g sensors, access logs data, and email data. There are many techniques and software available, which can process and provide efficient storage of structure and unstructured data and help organization to perform analytics on unstructured data. Unstructured data does not well-organized and not stored in predefined manner e.g. logs, web chats. So there is no common framework available to analysis the structure and unstructured data. In this paper the researcher has applied a common approach to analysis and provide the result in better way. Here mainly focus on open-source tools to analysis the data like elastic-search, Java, JSF, Apache tomcat etc.

Keywords: Kafka, Zookeeper, Fluent-td, Elastic search, Kafka-Connect, Stream, tomcat, JSF, Java.

.

1. INTRODUCTION

The massive size of both types of i.e., structured and unstructured data which is high in volume and difficult to process by using existing traditional computing technique. The importance of data for making significant business decisions is immense. An organization’s ability to gather correct data, interpret it accurately, and work on those insights is fundamental in determining its success. The key to unlocking the value of such massive amounts of data is understanding data structure. Data structure refers to a specific way of organizing and storing vast sets of data in a database or warehouse so that companies can access and analyze it quickly. However, organizations today are swarmed by the sheer amount of structured and unstructured data.

While structured data is relatively easy to search, unstructured data is more challenging to search, process, and understand.

The absence of a predefined model makes it challenging to deconstruct unstructured data. Further, unlike structured data, where there are multiple analytics tools available for analysis, there aren't many for mining and arranging unstructured data. In this paper propose a framework a common tool to analysis and construct the information from structured and unstructured data by applying custom filter on search.

In this research Fluent-td agent play very vital role to capture changes in log source files. Fluent-td agent is a distributed platform that turns your existing log files into event streams, so the any error or exception can see and respond immediately to higher admin or monitoring team. Fluent-td agent is built on top of Apache Kafka and provides Kafka Connect compatible connectors that monitor the generated logs of given directories. Fluent-td records the history of logs changes in Kafka topic, from where push back to persisted elastic database. Top of the elastic database develop the java tool to extract the data and parse according to the requirement of custom requirements and provide the meaningful information to the end user. This makes it possible for dashboard to easily visualize the exact status of any large cluster. Even if any issue or shutdown required to dashboard application required the all-streamed logs will persist in Kafka topic for that time and the connector will resume from the previous offset. So, the data lost chances are negligible. While applied the search filter on text on legacy RDBMS approach, it is very difficult to search. So the proposed solution approach will fulfill the requirement by applying the search on textual data.

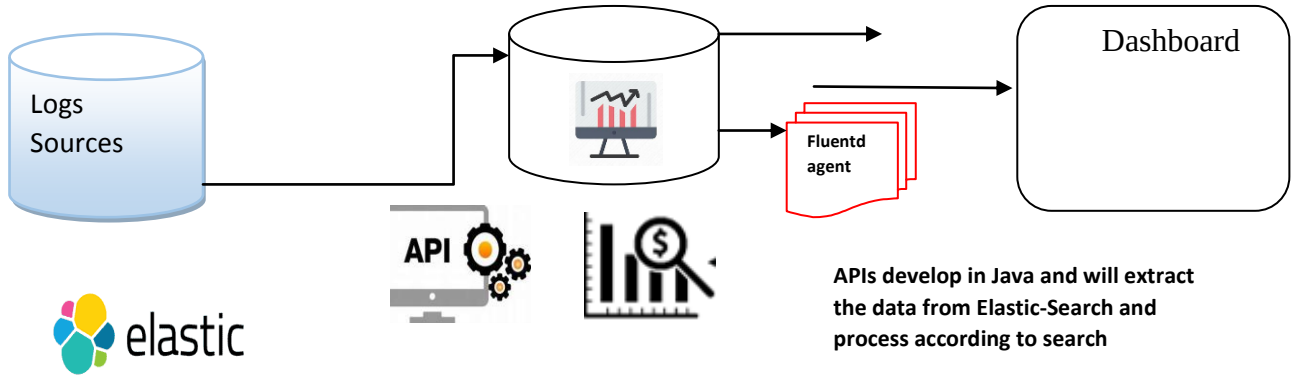

Figure1:Data Processing HLD

Till now very limited techniques have been developed to analysis and applied textual search onstructure and unstructured data at centralized mode. However, engineers of this field are scrupulously developing every day a new solution to extract the information from distributed and unstructured massive data. Till now there is no standard technique or architecture available to apply textual search on structured and unstructured data in a centralized analysis way.

Hence in the present study an attempt has been made by the researchers to design and develop a proficient solution to tackle the problems of textual search.

The aim of this architecture is to knob the analysis thedistributed unstructured data and to minimize the complexity of access and processing of large data. Designed architecture is used to successfully analyzed the existing datafrom elastic-search database.

In this architecture we are taking Kubernetes for deploy the container manager and elastic database to store the structure and unstructured data. Top of the elastic database develop a java-based API and JSF based GUI solution and deploy it on tomcat container to serve on web way.

The organization of present paper is in six different sections namely section-1: Introduction, Section-2: Related work, section-3: Data Analysis andInterpreted Challenges in big volume of data, section-4: Flow of Analysis and process Big Data, section-5: Conclusion and lastly paper ended with references.

Common API This API play vital role in this architecture. It is developed in java programming and will extract the data from elastic search indexes and process it and prepare this data to the meaningful information. Here are some steps to define the flow of this process tool.

i. Take input from user to extract the data from designed interface on web way.
ii. Fire the query to elastic search database and extract total count of the search and first slot of 100 records.
iii. Show these findings to the user. If user wants to get more detail, then again press next button.

Common API Controller the Controller take the input from the user and pass to the service layer, it is responsible to manage/map the user input request to desired services from service classes.

II. RELATED WORK

In present time, the digital universe comprises all structured and unstructured data spanning from videos, movies, surveillance video, photographs, data recorded through sensors, connected devices etc. and it is expected to increase up to 44 zettabytes by the 2020. Till now numbers of studies have been conducted in the field of big data at national and international platform. This section covers the related work conducted so far in this field as follows:

For continuing the part of literature review I have studied from official's site of tools that have I used in the proposed architecture.

LXC (Linux Containers) was the first, most complete implementation of Linux container manager. It was implemented in 2008 using cgroups and Linux namespaces, and it works on a single Linux kernel without requiring any patches.

Cloud Foundry started Warden in 2011, using LXC in the early stages and later replacing it with its own implementation. Warden can isolate environments on any operating system, running as a daemon and providing an API for container management. It developed a client-server model to manage a collection of containers across multiple hosts, and Warden includes a service to manage cgroups, namespaces and the process life cycle.

Included in 2013 as an open-source version of Google's container stack, providing Linux application containers. Applications can be made "container aware," creating and managing their own sub containers. Active deployment in LMCTFY stopped in 2015 after Google started contributing core LMCTFY concepts to lib container, which is now part of the Open Container Foundation. All those iterations had their adopters and devotees, but when Docker emerged in 2013, containers exploded in popularity. It's no coincidence the growth of Docker and container use goes hand-in-hand. Just as Warden did, Docker also used LXC in its initial stages and later replaced that container manager with its own library, libcontainer. But I've no doubt that Docker separated itself from the pack by offering an entire ecosystem for container management. With Docker, developers can create and run

application containers quickly. And with the release of Docker Hub, developers can download and run application containers even faster.

Kubernetes (Greek for "helmsman" or "pilot") was founded by Joe Beda, Brendan Burns and Craig McLuckie, was quickly joined by other Google engineers including Brian Grant and Tim Hockin, and was first announced by Google in mid-2014. Its development and design are heavily influenced by Google's Borg system, and many of the top contributors to the project previously worked on Borg. The original codename for Kubernetes within Google was Project Seven, a reference to a Star Trek character that is a 'friendlier' Borg. Kubernetes v1.0 was released on July 21, 2015. Along with the Kubernetes v1.0 release, Google partnered with the Linux Foundation to form the Cloud Native Computing Foundation (CNCF) and offered Kubernetes as a seed technology.

Kubernetes (commonly referred to as "K8s") is an open-source system for automating deployment, scaling and management of containerized applications that was originally designed by Google and donated to the Cloud Native Computing Foundation. It aims to provide a "platform for automating deployment, scaling, and operations of application containers across clusters of hosts". It supports a range of container tools, including Dockers.

Apache Kudu is a free and open-source column-oriented data store of the Apache Hadoop ecosystem. It is compatible with most of the data processing frameworks in the Hadoop environment. It provides a completes Hadoop's storage layer to enable fast analytics on fast data.

Fluentd is an open-source data collector, which lets you unify the data collection and consumption for a better use and understanding of data.Fluentd tries to structure data as JSON as much as possible: this allows Fluentd to unify all facets of processing log data: collecting, filtering, buffering, and outputting logs across multiple sources and destinations (Unified Logging Layer). The downstream data processing is much easier with JSON, since it has enough structure to be accessible while retaining flexible schemas.

From lastly three to four years, the amount of data streams has not stopped to increase in many fields such as financial applications, network monitoring, sensor networks and web log mining. Indeed, the Internet of Things accelerated the development of stream computing through the fast expansion of sensors deployed at geographically distributed locations Large Internet companies also shifted their workloads towards the stream model: for example, Facebook has to handle 106 events/s within a latency between 10 and 30s for advertisement purposes, which represents a data rate of 9GB/s at peak.

Le Paul Noac'h , CostanAlexandruInria,LucBouge´ ENS, emphasized how to detect the bottlenecks of Apache Kafka and how to fine tune its deployment. Researcher focus on optimizing the Big Data processing architecture for a given specific use-case. As the mechanisms of ingestion are better understood.

Much streaming data analysis using deep learning has been studied in recent years. Algorithms and architectures for high-speed and high-accuracy deep learning are being studied. However, these are premised on executing streaming data on a single computer. This research is different in that we consider the total analysis throughput, taking into account the data transmission between different-different sources to final point. In addition, because stream processing is performed using Spark Streaming, it is possible to easily perform extensions using the functions of Spark. Spark Streaming is applied to various technologies. The paper proposes DINAMITE, which is a tool kit that provides analysis tools implemented by Spark Streaming. The analysis tools of DINAMITE measure all

memory access with advanced debugging information and help programmers to identify memory bottlenecks. Chen and Bordbar use Spark Streaming as a solution to issues such as speed, scalability, and fault tolerance of rule-based systems that are currently used.

Ayae Ichinose,2 Hitotsubashi, Chiyoda-ku,Otsuka, Bunkyo-kupropose a video analysis framework that collects videos from multiple cameras and analyzes them using Apache Kafka and Apache Spark Streaming.In addition, it is confirmed that the number of cores is needed to consider for the efficient cluster configuration, and that the network bandwidth between the nodes becomes a bottleneck as the amount of data and the number of components increase. In this study the author mainly focuses on particular dataset not generic dataset.

Ayae Ichinose, Hitotsubashi, Chiyoda-ku, Otsuka, Bunkyo-ku propose a video analysis framework that collects videos from multiple cameras and analyzes them using Apache Kafka and Apache Spark Streaming. In addition, it is confirmed that the number of cores is needed to consider for the efficient cluster configuration, and that the network bandwidth between the nodes becomes a bottleneck as the amount of data and the number of components increase. In this study the author mainly focused on particular dataset not generic dataset.

III DATA PROCESSING/ANALYSIS CHALLENGES IN CONTINUOUS STREAMING

Every organization wants high availability information based on the dynamic requirements from various sources and in various formats. There is big challenge of data integration on central place to process and analysis prediction. To integrate the data from various sources and from various formats to central place there is architecture required which can provide a common solution and can fit any requirements. There is another challenge to streaming continuously and integrate legacy data which may be in different formats. After achieve the data in centric nature, there is another big challenge to process the data from big chunk. Due to the large volume of data, it is very difficult to process with legacy model, The proposed framework and API is in distributed in nature, It can be easily process the large volume of data in distributed processing and extract the desired result in minimum time frame.

The following key challenges of data processing

1) Volume of data are very huge.
2) Format of data are not well defined(unstructured)
3) Heterogeneous data input sources.
4) Dynamic user queries with multiple filters
5) Maintenance of Data (Live Streaming).
6) Legacy data.

To overcome these challenges there is a need to develop processing framework or any architecture. There are many open-source tools available in market to process and analysisdata likeApacheSpark, Apache Kafka, Kibana etc. With the help of these tools'authors developed architecture to process and analysislarge amount of data from different-different data sources to different-different data formats.

IV.FLOW OF ANALYSIS AND PROCESS BIG DATA

To process and analysis thehuge volume of data in big data environment firstly it is necessary to know about its formats. After identification of format of data, the following steps are to be follow:

Step 1: Take input from user for their requirement.

Step 2: Prepare the query string behalf of input parameters.
Step 3: Apply query and extract the total number of findings.
Step 4: Extract the first batch of records and prepare the result as per requirement.
Step 5: Display the result to user.
Step 6: If user required more result on same query then demand for next batch.

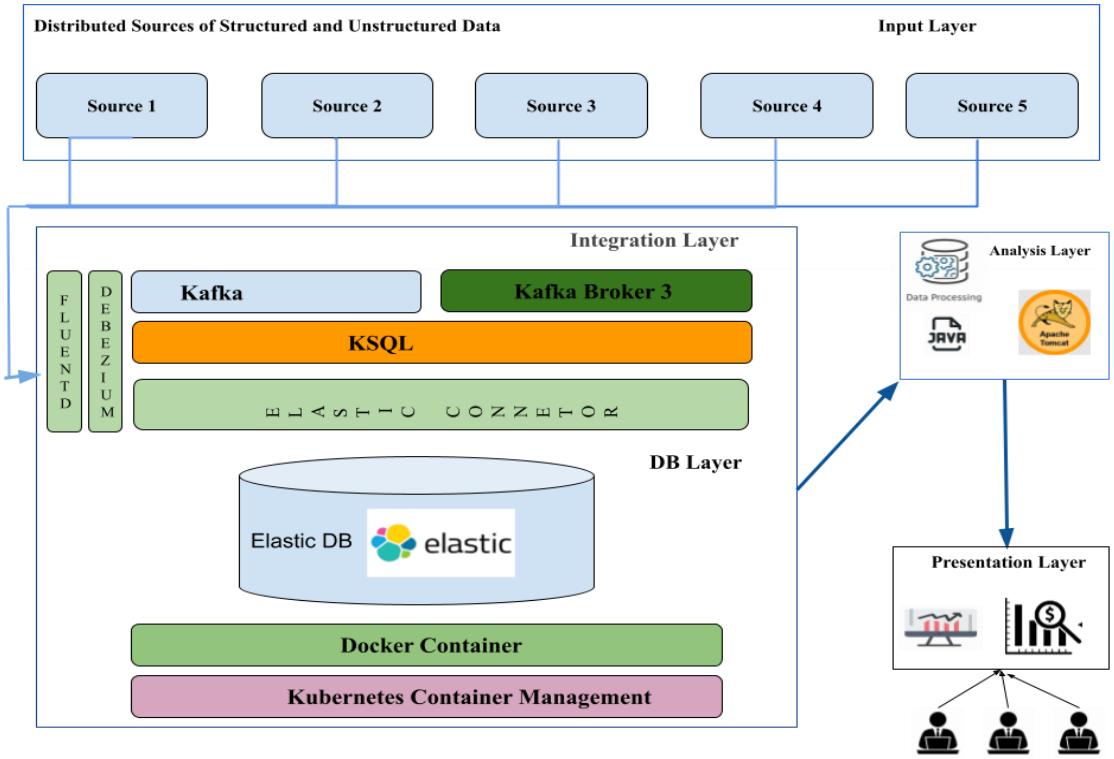

Figure 2: Data Analytics for Structured and Unstructured by Big Data Framework

In above Figure 2, The given architecture has been implemented. And it has been tested on more than 25 cr. (250000000) structured and more than 2 TB unstructured data.
The Summary of execution result for Structured data:

Sr.	Name of Relation	Size (no of record)	Processing with existing technique (RDBMS)	Processing time with proposed architecture (Java API)
1	Owner	180000000	3minutes	10 second
2	Tax	250000000	5 minutes	5 second

For Unstructured data:
Unstructured data have been processed 2 TB by using proposed architecture. The biggest benefit of this architecture is that user did not require to connect different – different sources to extract the information, only need to just install the proposed solutionand process the large amount of data from

central repository and data will start to come at dashboard. There is no other manual configuration required to make calculation or decision taken reports

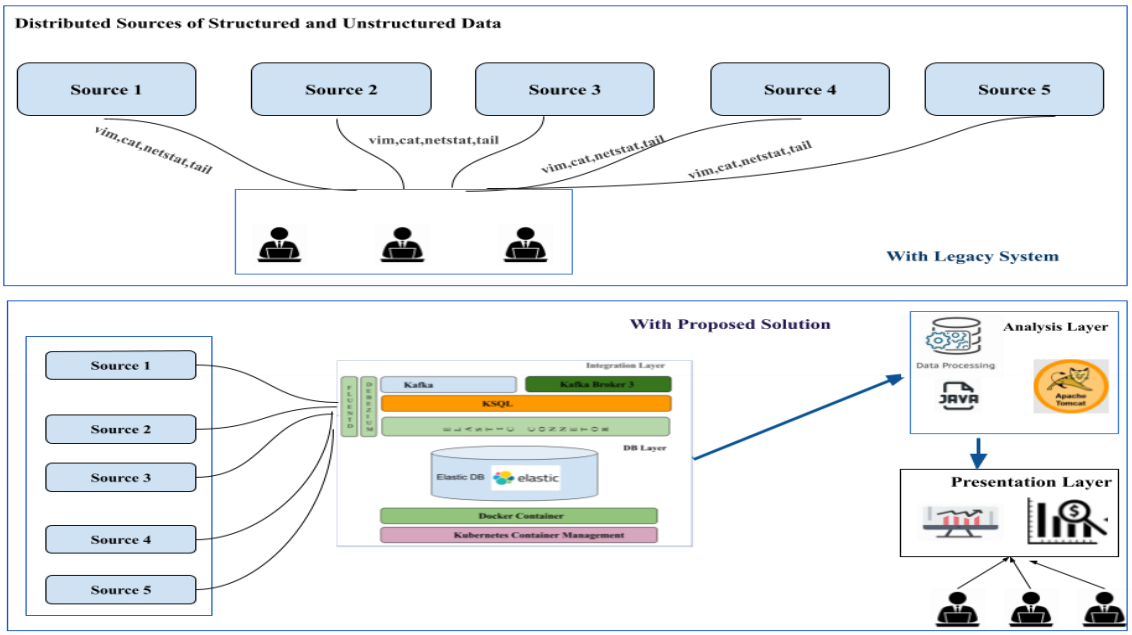

Figure 3: Comparison view between legacy and proposed architecture

Any anonymous or administrative user, those have little bit idea of the internal IT infrastructure of business, but they required information to take the necessary decision and observed the system from single window, And the second concern of any IT setup, you cannot allow multiple users to directly access the actual servers due to security reasons. The main concept of this architecture to avoid the direct access of actual machines and datafiles and directories. The target of the given architecture is to successfully process the data from different-different sources with different-differenttypes and provide the meaningful information to end user. In this architecture we are taking Haproxy access logs as unstructured dataset and processed them with use of elastic-search, Kafka and java APIs and provide the well define dashboard to meaningful reports.

V. CONCLUSION

In this study, it is showing thatin the digital era data is growing very fast. As Forbes report there are 2.5 quintillion bytes of data created each day at our current pace. Dataare collected in different – different formats such as access logs, systemlogs,sensor logs, IOT logs. To process those data and get meaningful information from it there is required to integrate on one central place and analysis/process it. The depended dataare generated from heterogeneous sources. Hence data integration is big challenge in current scenario. To ease of data integration process and analysis the data, authors are proposed a common solution to integrate and analyze the data from many sources to one place. In this architecture, the main focus on data analysis from integrated data and pull out the meaningful data sets from huge amount of data.In present study, the authors are applied the proposed solution on more then 25 cr. on structured data and more then 2 TB on unstructured data. The UI developed in JSF framework to take the user input to fetch the required information, and top of the integrated data there is a Java API developed to take the input from presentation layer and prepare

the query according to the elastic-search and fetch the desired result to fulfill the business requirements. For unstructured data in legacy era if any user want the information or any data report from many data sources that was very difficult to collect the information in real time. But the use of the proposed solution the business people can directly look and feel the real time data without any delay and use it in the business.

REFERENCES

[1] https://blog.aquasec.com/a-brief-history-of-containers-from-1970s-chroot-to-docker-2016.

[2] https://docs.docker.com/

[3] https://kubernetes.io/

[4] http://edit.dialogic.com/glossary/kubernetes

[5] https://github.com/kubernetes/kubernetes/

[6] https://www.wired.com/2015/06/google-kubernetes-says-future-cloud-computing/

[7] https://kudu.apache.org

[8] https://debezium.io/

[9] https://github.com/**debezium**/debezium

[10] Lu, Ruirui, Wu, Gang, Xie, Bin and Hu, Jingtong. "Stream Bench: Towards Benchmarking Modern Distributed Stream Computing Frameworks.." Paper presented at the meeting of the UCC, 2014.

[11] Milan ermk, Daniel Tovark, Martin Latovika and Pavel eleda, A Performance Benchmark for NetFlow Data Analysis on Distributed Stream Processing Systems, Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP

[12] Le Paul Noac'h , CostanAlexandruInria,LucBouge´ ENS "A Performance Evaluation of Apache Kafka in Support of Big Data Streaming Applications" 978-1-5386-2715-0/17 2017 IEEE.

[13] L. Qing, Q. Zhaofan, Y. Ting, M. Tao, R. Yong, and L. Jiebo, "Action recognition by learning deep multi-granular spatio-temporal video representation," in Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, ser. ICMR '16. New York, NY, USA: ACM, 2016, pp. 159–166

[14] H. Ye, Z. Wu, R.-W. Zhao, X. Wang, Y.-G. Jiang, and X. Xue, "Evaluating two-stream cnn for video classification," in Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, ser. ICMR '15. New York, NY, USA: ACM, 2015, pp. 435–442

[15] J. Read, F. Perez-Cruz, and A. Bifet, "Deep learning in partiallylabeled data streams," in Proceedings of the 30th Annual ACM Symposium on Applied Computing, ser. SAC '15. New York, NY, USA: ACM, 2015, pp. 954–959

[16] S. Miucin, C. Brady, and A. Fedorova, "End-to-end memory behavior profiling with dinamite," in Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, ser. FSE 2016. New York, NY, USA: ACM, 2016, pp. 1042–1046.

[17] Y. Chen and B. Bordbar, "Dress: A rule engine on spark for event stream processing," in Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, ser. BDCAT '16. New York, NY, USA: ACM, 2016, pp. 46–51.

[18] Ayae Ichinose, 2 Hitotsubashi, Chiyoda-ku , Otsuka, Bunkyo-ku "A Study of a Video Analysis Framework Using Kafka and Spark Streaming" 978-1-5386-2715-0/17 2017 IEEE

[19] Zhai G.,, Yang Y., Wang H., and Du S., (2020) 'Multi-Attention Fusion Modeling for Sentiment Analysis of Educational Big Data', Big Data Mining and Analytics, 3(4) pp. 311 – 319

[20] Tamer, Emara, Mahmud S., M., Zhexue J., Huang, Salloum S., Sadatdiynov K., (2020) 'A survey of data partitioning and sampling methods to support big data analysis', Big Data Mining and Analytics, 3(2) pp 85 – 101

[21] MishraSuyash, MishraAnuranjan (2017) 'Structured and Unstructured Big Data Analytics', International Conference on Current Trends in Computer, Electrical, Electronics and Communication, DOI: 1109/CTCEEC42587.2017