JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 07 Issue 01
2024

Research Article
https://doi.org/10.56979/701/2024

An Indexing-Based Architecture for Fast Data Retrieval in Smart Cities

Muhammad Bilal Aslam[1], Haiqa Mansoor[1], and Usman Akhtar[1,2*]

[1]Department of Computer Science, Riphah International University (RIU), Lahore, Pakistan.
[2]Faculty of Computer Science & Informatics, Berlin School of Business & Innovation (BSBI), Berlin, Germany.
[*]Corresponding Author: Usman Akhtar. Email: usman.akhtar17@gmail.com

Received: December 04, 2023 **Accepted:** April 28, 2024 **Published:** June 01, 2024

Abstract: A Smart City leverages Information and Communication Technologies (ICT) to enhance the quality of life for its residents by improving operational efficiency and providing reliable services. The primary objective of a Smart City is to use technology for optimization and decision-making. This involves deploying a network of sensors connected to the internet to collect real-time data from the environment, which is then shared across various systems to make intelligent decisions and improve city operations. Smart City applications encompass smart homes, transportation systems, traffic management, power consumption management, water supply networks, and other community services. The resulting vast amounts of data, known as Big Data, present significant challenges in terms of acquisition, management, and real-time analysis. Despite various proposed solutions by researchers, efficient data handling remains a critical issue. In our study, we propose an innovative architecture based on indexing to enhance data analytics for Smart City applications. This architecture covers data acquisition, storage, and retrieval. We validate our proposed architecture through the analysis of diverse datasets related to Smart Cities, demonstrating its effectiveness in managing and processing large-scale data efficiently.

Keywords: Smart City; Internet of Things; Urban Planning; Big Data; Data Analytics; Data Retrieval; Indexing; Indexing Based Framework.

1. Introduction

The change in technology from traditional desktop computing to undeniably modern computing along with the critical expansion in connected sensors and actuators have made it possible to live a smart life in a smart environment [1]. Researchers presented many smart setups like Smart Building & Home, Smart Transportation, Smart Power Consumption, Smart Water Management, Smart Waste Management, Smart Education (e-Education), Smart Governance (e-governance), Smart Medical Facility (e-Medical), and Smart Communications etc. All of them are grouped together and presented as a smart city. This area of study is becoming popular as it improves the citizen's life standard and makes them have everything available just from a single click.

Table 1. Research Challenges

Challenges	Description
Heterogeneity	While dealing with the Big data heterogeneity is the first challenge to be faced. Heterogeneity means variability in data. Different sensors generating data in different ways.

Normalization	Data is also denormalized which means data is out of range that leads to wrong decision. To improve the results data, need to be brought within a limited range.
Noisy Data	From this bulky data, valuable data is difficult to find out. Noisy data needs to be removed to infer useful decisions.
Data Format	Different sensors produce data in different formats. These formats may vary in semantics, type, shape and representation. One sensor is producing data in digital form and the other may be in analog form.
Missing Values	Sometime data is incomplete that contains some missing values that may mislead the actual results.
Data Retrieval	To make smart decisions there is a need for fast and desired data retrieval. Retrieving the desired and useful data from the immense data is a challenging task.
Data Volume	Data size is a big challenge to handle and deal with.

By looking at research gaps we choose data retrieval domain. For that we introduced an architecture for effective data retrieval in smart city planning. The main purpose of the proposed architecture is to introduce a system that maximize the data retrieval with minimal time. As the data need to be retrieved as early as needed.

Technically, this click is not as simple as to say, thousands or even millions of sensors, cameras, actuators and smart meters must be used to make a city as smart city. Apart from managing these hardware components, these also generate the data as many of these hardware components i.e. sensors and meters will go on working 24/7 resulting in generating bulky data that is too difficult to handle. This bulky data referred to as big data. To differentiate data from big data we use 3 Vs i.e. volume, velocity, variety [2]. This data is gathered from devices, then stored and after that analysis is applied, to find the insights to deduct useful results. Big data have various applications in social media analytics, government, fraud detection, call center, smart city etc.

Big data systems are used for accusation, storage, process and analysis of data in smart cities to improve the services of smart cities. Apart from that it also helps what changes and expansion can be made in smart city services. This expansion is never ending, however big data can help to make advance services in smart cities using appropriate tools for efficient data analysis. This efficiency will encourage collaboration and communication between different entities of smart city and help to add some extra services to smart city. So, this will serve and help to improve business [2]. Numerous sensors, smart appliances, and actuators generate huge amounts of data on a regular basis, and extracting relevant information while retaining acceptable processing speed, optimal aggregation of computing performance to enhance query execution and data search performance has become crucial. Here our research study is focused on the retrieval rate. Data retrieval rate needs to be improved. Various approaches have been explained that are used by different researcher for smart city planning. Different architectures are discussed. Here, we also introduced an architecture for effective data retrieval in smart city planning. The main purpose of the proposed architecture is to introduce a system that maximizes data retrieval with minimal time. We proposed a four-layer architecture with: Data Acquisition, Data Storage, Data Retrieval, Data Visualization.

Since the smart city is very vast area of study as thousands or even millions of devices and sensors are deployed from smart homes to health care, from e-learning to smart transportation, from smart parking to smart security, from smart water consumption to smart sewerage, from smart garbage collection to smart power consumption so on and so forth. Every component of a smart city has thousands of sensors and meters that are 24/7 continuously generating data. The data with this kind of volume is termed Big Data. While dealing with this kind of data we have to face different issues related to data. Some of those issues are:

In this study, we focus on the domain of data retrieval within Smart City planning. Our primary goal is to introduce an architecture that optimizes data retrieval processes, ensuring minimal retrieval time and maximizing efficiency. Given the complexity and scale of Smart City environments, where millions of sensors and devices operate continuously, it is imperative to develop robust systems capable of handling and processing large-scale data effectively.

The proposed architecture comprises four layers: Data Acquisition, Data Storage, Data Retrieval, and Data Visualization. This layered approach aims to streamline the entire process, from data collection to actionable insights, thereby enhancing the operational efficiency of Smart Cities. Our architecture leverages indexing techniques to improve data retrieval rates, ensuring timely access to relevant information, which is essential for real-time intelligent decision-making and autonomous data aggregation.

The rest of the paper is organized as follows: Section 2 provides an overview of recent work in the fields of Big Data analytics and IoT for smart city management. The open research questions and challenges are highlighted in Section 3. The proposed architecture is described in Section 4 of the paper. Section 5 contains the analysis and outcomes. Section 6 specifies the conclusion.

2. Recent Literature

The continued expansion of smart cities diverts researchers' attention towards a good architectural design. The conventional smart city architecture can provide researchers with several benefits. The smart city also covers a wide range of research methodologies associated with IoT and Big Data analytics, from conceptual to a complete set of activities. In recent years, research groups have been working on a new approach to describe the overall structural architecture for smart cities using IoT and Big Data analytics. Similarly, several recommendations have been made in the literature that seek to solve the problems through extensive research and computations based on test beds.

To evaluate the potential advantages of Big Data analytics for smart cities M. Babar et al. [2] The new development and extension in the field of Internet of Things (IoT) is giving an extraordinary business imminent toward the new era of smart city. The knowledge of the smart city is broadly liked, as it improves the greatness of life by smart residents, transportation, smart health care, smart weather forecasting, medical care, etc. Consistent escalation of the complex city set-up is widely tested by constant handling of information and smart choice capacities. Thus, in this paper, they have proposed a smart city design which is in view of Big Data Analytics. The proposed conspire is included three modules: (1) data procurement and accumulation module gathers different information interrelated to city administrations, (2) data processing and handling module performs standardization, filtration, preparing and data investigation, and (3) application module formulates decisions and initiates events. The proposed design is a conventional answer for the smart city arranging and assortment of datasets is investigated to approve this design. Moreover, they tried dependable datasets on Hadoop worker to check threshold limit value (TLV) and the examination shows that the proposed conspire offer important approaching into the local area improvement frameworks to improve the current smart city engineering. Also, the effectiveness of proposed engineering as far as throughput is likewise shown

S. Ameer et al. [3] numerous challenges to be faced while managing and analyzing IoT data due to rapid increase in connected devices. This rapid increase in connected devices causes the necessity of architectures to process this huge volume of data. This paper proposed an architecture based on Apache Spark and Hadoop for real time data processing. This architecture aims to manage and analyze the smart city data on pollution dataset.

The proposed architecture in the paper composed of four layers:

i) Data Gathering
ii) Communication
iii) Data Management
iv) Application Layer

The top layer data gathering is responsible for collecting data from different sensors that are used in smart city. Since data from different devices is collected in different formats, there is need to pre-process and aggregate it. So, both these tasks are done in this layer and after that the data is sent to the communication channel to store this data so that further processing may take place. The next layer i.e. Communication Layer contains technologies like 3G, 4G, LTE, Wi-Fi, ZigBee etc. All the data transferred from previous layer to data storage layer by using these available technologies. This layer acts as a gateway.

Latency rate can be improved using Fog computing. Just after the next layer is Data Management and Storage layer. The primary function of this layer is to store and analyze the data. This architecture is designed for online data processing, so they have used third party tools like Spark, VolitDB and storm. Data is stored on HDFS system. In data analysis different prediction patterns are also predicted. The last layer i.e. Application layer relates to different devices. These devices may get real time data to infer decisions.

To validate this architecture, it is implemented using a pollution data set of Aarhus City. Processing performance is also calculated using Python Data frames and Spark RDD and it has been observed that the processing time of Spark is about 10 times faster than of Python Data frame. They have claimed that their architecture is quite better than those of the traditional programing paradigm and it's true as they have proved it by implementing using different datasets. Moreover, the traditional programing paradigm are not quite well to handle big data. Future direction is that the same architecture can be implemented using various other machine learning algorithms that may improve efficiency, data retrieval rate as well as reduce error rate.

T Lakshmi et al. [4] Hadoop Distributed File system is fault tolerant and inexpensive hardware installable Apache Hadoop based project which is used to handle processing of big data. It is used for data storage, retrieval and data processing. This paper measured the performance regarding the read write operation for both small as well as large files. They have used an open-source project Hadoop. The results show that files that are larger in size than those of the default block size have high performance while the files that are below from the default size have poor performance.

The test environment is a five-node cluster with 2 GB RAM, Intel Core 2 Duo Processor of 2.9 GHz and 500 GB SATA Hard Disk. The test operating system is Ubuntu-12.04 with kernel version 3.2.1. The used Hadoop version is 0.20.0 with JDK 6. As we are only focused on reading operation so here, we discuss only write operation in the paper. It's been observed that HDFS will take more time to read the data when the files use the block size which is less than that of the default block size.

B. Nathali Silva et al. [18] in the recent era the Smart City has become a buzzword. Although many researchers worked on different projects like water reservation system, smart parking, smart power consumption, smart education, e healthcare etc. but still its implantation as a project Smart City is immature. Many reasons are for that like complex urban network, gigantic amount of data from different sources, data processing and decision-making capabilities.

This paper presents architecture with three levels.

1. Data Generation and Acquisition Level (Gather data from different heterogeneous sources)
2. Data Management (This involves filtering, data analysis and storing)
3. Application Level (User may interact to make useful decisions)

According to B. Nathali Silva et al.[18] when the data is fuzzy and it also include facts and figures in raw form, so before storing the data in HDFS, first we have to filter out the noisy and unnecessary data using KF filter. This helps to reduce the processing time and improve the efficiency of the system [10]. After processing data is stored, Distributed manner using HDFS. This layer has a connection with the Application Layer. The main discussion in this paper is to compare the efficiency in terms of processing time with and without filtering the data before storage. It is proved by experimentation that by data filtration and removing the unnecessary data reduce the processing time. The proposed system is compared with Hadoop single node and Java query system. Data fusion property is used to improve the processing speed by removing unnecessary data. This study is focused on dealing with specific issues, but it may be generalized in future.

3. Proposed Architecture

In order to develop smart city, several sensors, actuators, smart cameras and meters are deployed. So, data from these sources is gathered and integrated with architecture. The key concept behind the proposed architecture is to improve the data retrieval rate in order to provide the right information at right place in right time using information and communication technology (ICT). This will help the citizen and authorities to infer useful decision. The volume of the gathered data is so tremendous that tradition databases are inefficient to handle this data. [2] Hadoop distributed file system is used for data storage. Indexing is used for data retrieval [5] and after that data is visualized in the form of charts. The details of each layer of the proposed architecture is explained in the upcoming sections

.

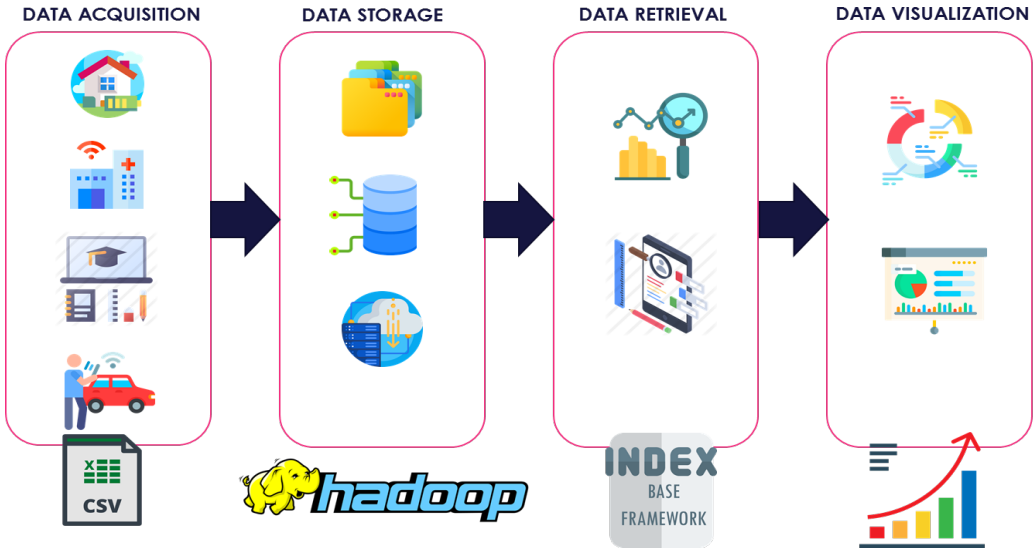

Figure 1. Block Diagram of the proposed approach.

3.1. Data Acquisition

Since the smart city is very vast area of study as thousands or even millions of devices and sensors are deployed from smart homes to health care, from e-learning to smart transportation, from smart parking to smart security, from smart water consumption to smart sewerage, from smart garbage collection to smart power consumption so on and so forth. Every component of a smart city has thousands of sensors and meters that are 24/7 continuously generating data. The data with this kind of volume is termed Big Data.

Before data storing, there is need to preprocess it so that affective analysis is carried out. The primary goal of data preprocessing is to remove unnecessary data so it may not deal as garbage and not produce misleading results. Preprocessing is also made to remove null values. Null values are present in the data due to different reasons like during a survey people may not answer some questions. Sometimes data preprocessing refers to data normalization that means data needs to be brought to a defined range. Different techniques may be used to normalize the data. Min-Max is one of the most popular technique used for normalization. [2] Suppose we have a value X that need to be normalized as Y in the specified range of [m-n]. This can be done using following formula:

$$Y = \frac{X - Xmin}{Xmax - Xmin} \times (n - m) + m$$

Here,

X is the value to be normalized

Y is the value obtained after normalization

X_{min} is the minimum value in the corresponding data block

X_{max} is the maximum value in the corresponding data block

m is the minimum value of the range

n is the maximum value of the range

The algorithms to implement the Min-Max Normalization is as under:

Procedure **Min-Max Normalization** *Procedure*

BEGIN:

INPUT: = *value_to_be_normalized*
min_value_of_data_block
max_value_of_data_block

SET: = *range_value [m-n]*

DO $Y = \frac{X-Xmin}{Xmax-Xmin} \times (n - m) + m$ (for each value to be normalized)

OUTPUT: = *normalized_value*

END

In spite of data normalization, lot of other preprocessing also required depending upon the nature of data but here the main focus of study is to improve the data retrieval rate, so we assumed that all this task

of data acquisition along with data preprocessing is done by development department of smart city. This department is responsible for deployment of different sensors, actuators and smart cameras etc. along with data acquisition and data preprocessing.

3.2. Data Storage

The second layer in our proposed architecture is data storage. After the data acquisition and preprocessing, data needs to be stored somewhere so that further processing may take place to deduct result. Since many sensors are deployed in smart city and they are continuously generating peta bytes of data that referred to big data. Traditional data storage platforms are not sufficient to handle this data. There are many issues in these platforms that make them not reliable to handle this data. Some challenges faced while using traditional databases to handle Big data are as under:

i. Huge Data Volumes (Huge amount of data)
ii. Nature of Data (Semi structure or unstructured data)
iii. High Data Velocity (How fast data is generating!)

To avoid all the above and some other challenges a distributed file system named Hadoop is used in this study. Hadoop is used for its reliability and economic benefits. One most important thing about Hadoop is that it uses a powerful map-reduce mechanism. Inspite of the single processing system that have a sole processor along with cache, RAM and ROM etc. Parallel processing is done using specialized hardware or in some cases utilized the existing hardware in such a manner, by dividing the task into modules along with parallel processing. This parallel processing is done using nodes. Distributed Files is mostly used when files are in larger size. So, the file is divided into small chunks (default 64MB).[4]

The association between these chunks and location of these files is stored in a separate file called name node or master node. A directory is also there which keeps track of the storage location of all these files. Researchers and practitioners also encourage this to store data in the form of chunks rather than storing data. Every big data processing software has its own mechanism to create data blocks and store those in a distributed manner. By default, block size is fixed and may be changed according to requirements and nature of data. Where storing data in blocks in the distributed form has many advantages, it also increases the data retrieval time. To overcome this an index-based architecture is explained in next section.

First of all, blocks are created. After that data is filled in the block until data reach the maximum storage capacity. Suppose D is the dataset that have x number of records, n number of blocks are created from this dataset.

$$D = \sum_{i=1}^{x} \square\, Record\ i \quad (1)$$

Here D represent the dataset that have x records

$$B(n) = \sum_{j=1}^{k} \square\, Record\ k \quad (2)$$

Here n number of B (Blocks) created with k records in each data block. Here we use the Hadoop distributed file system where default block size is 64 MB. Once blocks are created, data uploaded in those blocks with their replicas.

The algorithm for Blocks Creation is as under

Procedure **Data_Block_Creation**

```
BEGIN:
block_size ← DEFAULT
block_capacity ← TURE
total_block ← 0

DO
        IF (block_capacity ← TRUE) then
            add_record_in_block
        ELSE
            total_block ← +1
            set block_capacity = TRUE
        END IF
WHILE (end_of_file ← !NULL)
END
```

3.3. Data Retrieval

The third layer in our proposed architecture is data retrieval layer. After the data storage there is a need for data retrieval, so we may deduct some useful insights from the stored data. Data retrieval is the focus of this study. Data retrieval is important as we have to take lot of decisions on the base of retrieved data. [4, 5] Data need to retrieve as early as possible after query execution. Delay in data retrieved is meaningless because in smart city lot of decisions have to be on spot. So, improvement in data retrieval time is the need of the hour. This study focuses on this this research challenge.

Lot of researchers address this issue in different manners and proposed different techniques and schemas to solve this issue. They have proposed different techniques with different technologies. So, here we have proposed a technique along with algorithms that are the fine contribution to solve this research challenge as in today's world where everyone has limited time, wrote a query and then waited for response make people frustrated. Especially in case of smart city, when there is Big data, that is difficult to process due to its huge volume, velocity and verity. So, maintaining the computing performance while extracting useful data is critical. There is a need to design a solution that speeds up the query and fetch the information with less time than the existing frameworks. Index-based retrieval minimal the data retrieval time. Detailed working is explained in the rest of the sections.

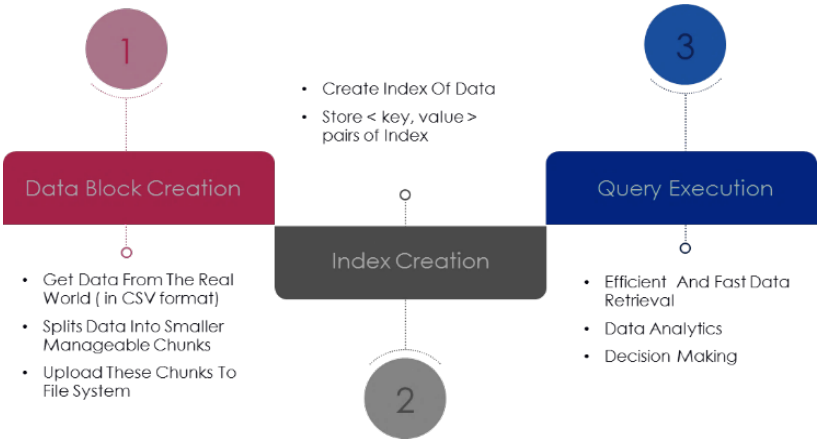

Figure 2. Key Feature of Architecture

After data is uploaded to blocks, indexes are created. The primary purpose of index creation is to minimize the data retrieval time. A separate index is created for each block. Indexes are created by taking data from each record. Below is the algorithm to create the index. Here index attribute is taken as key and location of record as a value and store it in BTREE as illustrated in algorithm.

Procedure **Index_Creation**

```
BEGIN:
index_clmn ← NULL
DO
    input:index_clmn
    FOR (dl_index_clmn)
        DO
            create empty BTREE
            WHILE (index_clmn_data ←!NULL )
                BTREE ← add <key, value >
            END WHILE
    END FOR
WHILE ( index_clmn ←!NULL)
END
```

Last task is the query execution. Based on this we can validate our system performance. Before executing the query, the compiler checks the syntax of the query. If there are some mistakes in the syntax

system will display the message to check and correct the syntax. After that on the base of the written query, data is displayed along with query execution time. From that time, we can validate our system.

Procedure **Query_Execution**

BEGIN:
INPUT: $= Query$
IF $(check\_syntax \leftarrow FALSE)$
$display\_error$
ELSE
$get\_provided\_files\_name\_from\_query()$
$get\_desire\_data()$
$display\_data()$
$display\_time()$
END

4. Results and Discussion

This section provides implementation of the proposed architecture as well as results obtained using this architecture. Analysis is performed using different datasets in the proposed architecture. Datasets are obtained from different valid and authentic sources. [19] Validity of the proposed architecture is check by implementing different datasets. The effectiveness of the system can be seen from the results.

4.1. Test Datasets

Explicitly available datasets are used which are reliable and authentic. Multiple data sets like Road Traffic Data, Parking Data, Pollution Data, Water Meter Data are discussed and proposed architecture is validated using these datasets. Description of each data set is given in the below table. It contains the information of each dataset like size of datasets, no. of records as well no. of attributes in each dataset.

Table 2. Datasets Attributes

Datasets	Data Size (KB)	No. of Records	No. of Attributes
Road Traffic Data	102	450	26
Parking Data	3648	55265	6
Pollution Data	79114	17770	8
Water Meter Data	10916	71730	11

Detailed description about each dataset is also explained:

Table 3. Datasets Description

Dataset	Description
Road Traffic Data	This dataset include data for the six months period. The data involve the vehicle traffic data between two points. The file format is CSV. The dataset includes attributes like starting point as street name, duration in seconds, city, distance in km, exit Id, postal code and some other attributes. [19] Link: http://iot.ee.surrey.ac.uk:8080/datasets.html#traffic

Road Parking Data	This data is taken from the city of Aarhus. Although, this data is simple but have huge amount of data. It involves the vehicle count, update time, id, total spaces, garage, code, stream time attributes. This dataset also has the information of 8 parking slots of Aarhus city for duration of six months. File format of this dataset is also CSV.[20] Link: http://iot.ee.surrey.ac.uk:8080/datasets.html#parking
Pollution Data	Where we talk about the traffic there definitely pollution. So, we also take the traffic dataset and deduct some insights that make city smart city to control pollution. This CSV dataset also having fewer attributes. The attributes include ozone, particulate matter, carbon monoxide, Sulphur dioxide, nitrogen dioxide, longitude, latitude and time stamp. This data is collected using air quality index.[21] Link: http://iot.ee.surrey.ac.uk:8080/datasets.html#pollution
Water Meter Data	Water meter dataset includes data about different installed water meters. Water meters are the meters that used to calculate the amount of water that flow through the water connection. It includes facility id, House No., Street No., Account No., Status, GPS, Image Link. [22] Link: https://data.surrey.ca/dataset/water-meters

4.2. Evaluation Metrics

Evaluation metrics is the essential component of this study. Evaluation metrics are the factors that need to be assessed during the study. Here in this study query execution speed and after query execution, data retrieval rate is evaluated. Proposed architecture is implemented and evaluated using dataset of different natures. A comparison analysis is made with existing approaches. Existing approach data retrieval time and proposed approach time is measured and huge difference is found. Different queries are executed to filter out different data to validate the system and, in each case, quite good results are found that must be discussed in later section of this chapter. As our study is focused i.e. to improve the data retrieval rate, to retrieve more useful results within less time. So, I our case only evaluation matric is one and it is measured in term of time with unit second [38].

4.3. Experimental Environment and Implementation

In this section we will explain the hardware and software requirements to implement this architecture. This study is implemented using single node Hadoop cluster on Linux Operating System using map-reduce mechanism. In hardware, Core i3 processor with 4GB RAM is used. Below table contain the hardware and software specifications used in this experimental environment.

Table 4. Experimental Environment

Sr.	**System Parameter**	**Values**
1	Operating System	Ubuntu 20.04 LTS
2	Processor	Intel core i3 – 1.7 GHz
3	HDD Volume	20 GB
4	RAM	4 GB
5	File System	Hadoop Distributed File System
6	Data Warehouse Framework	Hive
7	Query Language	HQL (Hive Query Language)

5. Results

It's been observed that for different data sets say Pollution dataset while we retrieve the data where ozone value is less than 100 it filters out 10190 records in both the cases either query is executed without applying indexing to ozone column or with indexing. This show the search result or the revived data volume is same. But while we execute the query before applying indexing, the data is fetched after 3.99

seconds and after applying indexing it takes 1.239 sec to retrieve the same data. This means 69% time is saved by the user to get the same results.

But in another case when we retrieve the data with less than 100 amounts of nitrogen_dioxide then 11% time is saved for the user to access the same number of records. This shows that the data retrieval time using indexing is always less. In some cases, it's quite good and in some case, it is not as good but still it makes faster access to the data in a comparison to data retrieval without indexing. This is also shown is graph.

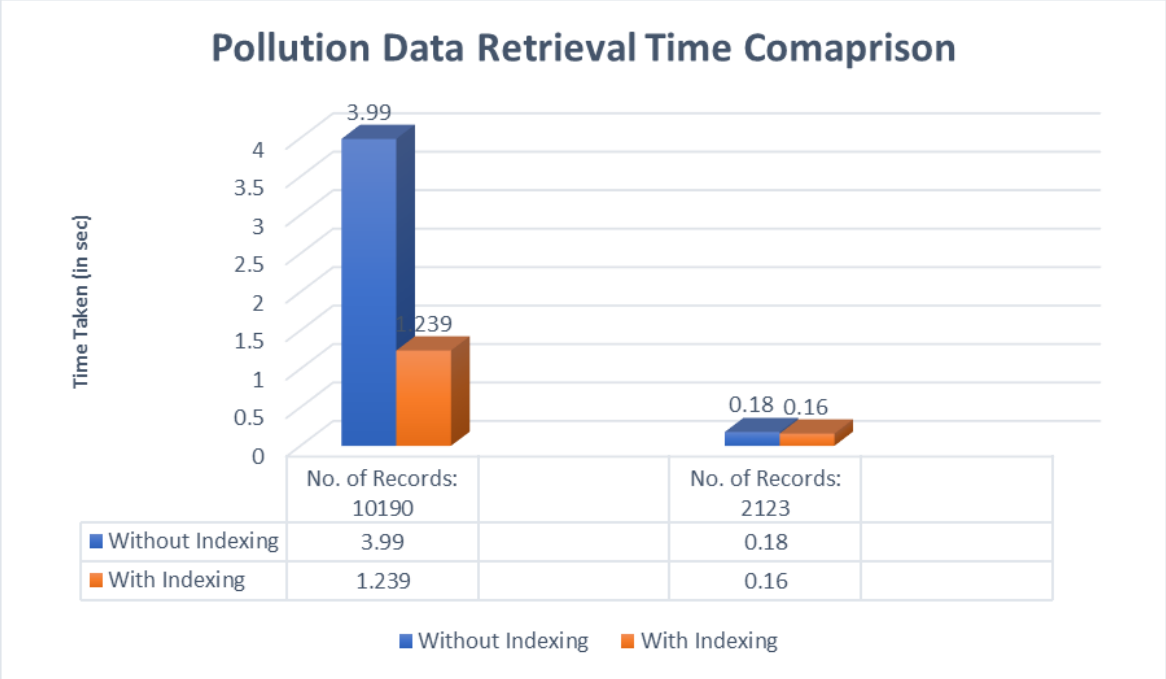

Figure 3. Pollution Data Retrieval Time Comparison

In another case let's talk about the water meter dataset while we retrieve the data where "sname='TOWNLINE DIV' ", it filters out 20 records in both the cases either query is executed without applying indexing to "sname" column or with indexing. This show the search result or the revived data volume is same. But while we execute the query before applying indexing, the data is fetched after 1.497 seconds and after applying indexing it takes 0.295 sec to retrieve the same data. This means 80% time is saved by the user to get the same results.

For the same data set while execute another query when GPS='N'; we retrieve 2928 number of records withing 0.435 without indexing and after indexing same number of records within 0.186 sec. This shows that the data retrieved using indexing is always less. In some cases, it's quite good and in some case it's not as good but still it makes faster access to the data in a comparison to data retrieval without indexing. This is illustrated using graph.

Similarly, in traffic dataset whilePOINT_2_CITY='Aarhus'; it filters out 291 records in both the cases either query is executed without applying indexing to POINT_2_CITY column or with indexing. This show the search result or the revived data volume is same. But while we execute the query before applying indexing, the data is fetched after 0.28 seconds and after applying indexing it takes 0.17 sec to retrieve the same data. This means 39% time is saved by the user to get the same results [35] [37].

While executing another query in that case when we retrieve the data while DURATION_IN_SEC<100; then 9% time is saved for the user to access the same number of records. This shows that the data retrieved using indexing is smaller. In some cases, it's quite good and in some case it's not as good but still it makes faster access to the data in a comparison to data retrieval without indexing. Below graph also explains the same [35] [36].

While working with parking dataset and execute the query select * from parking_tab_managed where garagecode="SALLING"; It displays 10190 record in each case whether it works with indexing or not with a retrieval time 3.99 sec without indexing and 1.239 sec with indexing as shown in below graph:

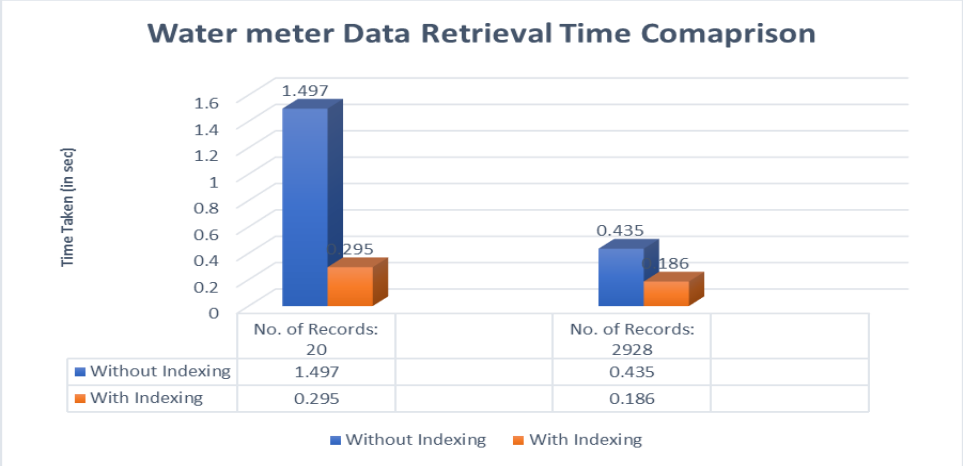

Figure 4. Water meter Data Retrieval Time Comparison

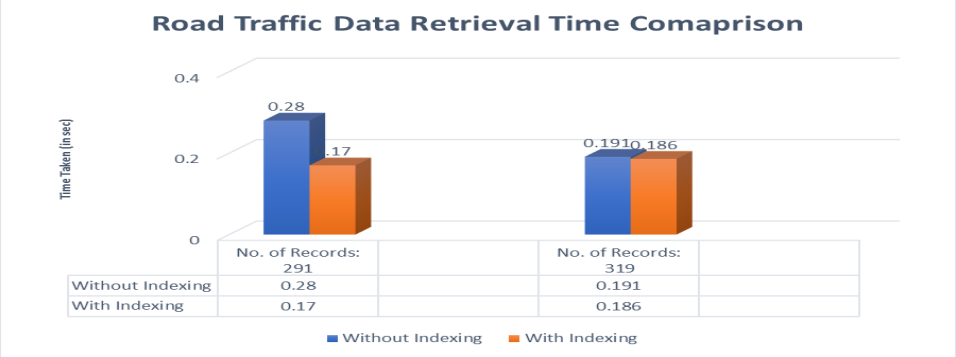

Figure 5. Road Traffic Retrieval Time Comparison

.

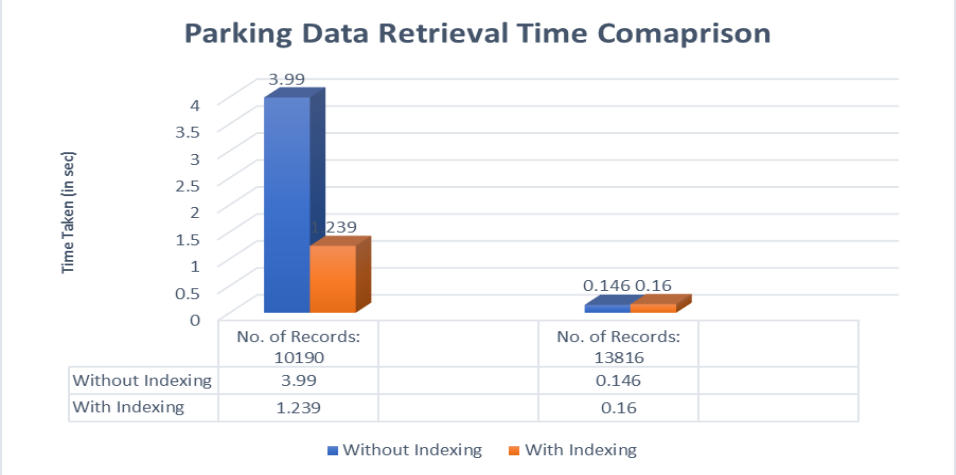

Figure 6. Parking Data Retrieval Time Comparison

6. Discussion

An indexing-based architecture is presented to improve the search performance and query execution speed for large volume datasets. [5] The presented architecture focuses on the volume, velocity and verity of the large datasets. This architecture has various modules data acquisition, storage data retrieval and data visualization. We use variety of datasets in our experimentation and the results show that data retrieval time of a query with indexing is quite less than the query exaction without indexing. Indexing make faster access to data keeping same number of search result as without indexing. It is efficient for both small as well as large datasets [33] [34].

However, the performance may vary with the dataset schema like number of records and number of attributes etc. Index based architecture actually is the blend of three module one is the block creation, its primary goal is to divide the whole data into chunks named as data blocks and store them is distributed manner. Second, is the index creation, it is to create the index of the desired column and the third is the final module, and it is used to execute the query and to fetch the desired results. Data retrieval time using indexing is always less. In some cases, it's quite good and in some case, it is not as good but still it makes faster access to the data in a comparison to data retrieval without indexing.

7. Conclusion and Future Work

This research provides a comprehensive understanding of the role of big data in smart cities, focusing on the enabling technologies and future business models and architectures for managing big data. We also examined various smart city applications where big data analytics can play a pivotal role, supported by relevant case studies. Additionally, we identified open research problems to guide future scholars in this domain. While big data holds significant potential for extracting valuable insights and supporting decision-making processes, research in this area, particularly in the context of smart cities, is still nascent and presents numerous challenges that need to be addressed.

The primary objective of this study was to develop an intelligent decision management and control center that facilitates efficient data collection, storage, retrieval, and analysis. By evaluating diverse datasets, we demonstrated how big data can be leveraged for future smart city development and planning. Our proposed approach, though targeted to specific objectives, does not offer a universal solution for all smart city systems. However, it includes a scalability option to extend the current work in the future. This study focused on particular smart city challenges to create a sophisticated environment for data testing. We validated our proposed smart city architecture using a real-world dataset from Aarhus, Denmark, showing that the indexing-based framework is efficient for rapid data retrieval and decision-making.

In future work, we plan to conduct simulated experiments to verify the precision and efficacy of the proposed framework. Additionally, we aim to evaluate our scheme against smart urban architecture standards. Future research will explore integrating B-tree indexing with probabilistic machine learning methods. B-tree indexing, which uses a tree structure, will be employed alongside probabilistic modeling to create adaptive indexes by anticipating future query workloads and dynamically adjusting index attribute sets. Furthermore, we will implement disaster recovery mechanisms to ensure fault tolerance and data restoration capabilities in large-scale distributed storage systems.

By advancing these aspects, we aim to enhance the robustness and efficiency of data management and retrieval processes in smart cities, contributing to the creation of more responsive and resilient urban environments. We're also putting Disaster Recovery [31] [33] in place to ensure that huge data distributed storage systems are fault-tolerant and that they can be restored.

References

1. Hashem, I.A.T., et al., The role of big data in smart city. International Journal of information management, 2016. 36(5): p. 748-758.
2. Babar, M. and F. Arif, Smart urban planning using Big Data analytics to contend with the interoperability in Internet of Things. Future Generation Computer Systems, 2017. 77: p. 65-76.
3. Ameer, S. and M.A. Shah. Exploiting big data analytics for smart urban planning. in 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall). 2018. IEEE.
4. Krishna, T.L.S.R., T. Ragunathan, and S.K. Battula. Performance evaluation of read and write operations in hadoop distributed file system. in 2014 Sixth International Symposium on Parallel Architectures, Algorithms and Programming. 2014. IEEE.
5. Siddiqa, A., A. Karim, and V. Chang, SmallClient for big data: an indexing framework towards fast data retrieval. Cluster Computing, 2017. 20(2): p. 1193-1208.
6. ITU. ITU-T, Smart Sustainable Cities at a Glance. Available from: https://www.itu.int/en/ITU-T/ssc/Pages/info-ssc.aspx#:~:text=%E2%80%8B%E2%80%9CA%20smart%20sustainable%20city,generations%20with%20respect%20to%20economic%2C.
7. How has New York City developed as a smart city? Evaluating smart city contributors in New York City Available from: https://docplayer.net/220237930-How-has-new-york-city-developed-as-a-smart-city-evaluating-smart-city-contributors-in-new-york-city.html.
8. Hollands, R.G., Will the real smart city please stand up? Intelligent, progressive or entrepreneurial? City, 2008. 12(3): p. 303-320.
9. Sazonchik, V., From smart technologies to smart cities. Smart city strategy, 2018.
10. Madakam, S., Internet of things: smart things. International journal of future computer and communication, 2015. 4(4): p. 250.
11. Ahmat, M.A., Big Data meet the eyes of the librarian. 2013.
12. Lauzon, D., Introduction to Big Data. 2012.
13. Sigma Computing. Available from: https://www.sigmacomputing.com/blog/top-20-big-data-statistics/.
14. Roy, A.S. How facebook handles the 4+ petabyte of data generated per day! ; Available from: https://medium.com/@srank2000/how-facebook-handles-the-4-petabyte-of-data-generated-per-day-ab86877956f4.
15. Thusoo, A., et al., Hive: a warehousing solution over a map-reduce framework. Proceedings of the VLDB Endowment, 2009. 2(2): p. 1626-1629.
16. Zeebaree, S.R., et al., Characteristics and analysis of hadoop distributed systems. Technology Reports of Kansai University, 2020. 62(4): p. 1555-1564.
17. What is Data Retrieval? - Definition from Techopedia. Available from: https://www.techopedia.com/definition/30140/data-retrieval.
18. Nathali Silva, B., M. Khan, and K. Han, Big data analytics embedded smart city architecture for performance enhancement through real-time data processing and decision-making. Wireless communications and mobile computing, 2017. 2017.
19. CityPulse Smart City Datasets - Datasets.[]
20. Road Parking, C.S.C.D.-. Datasets, Editor.
21. Datasets, C.S.C.D.-. Pollution Dataset.
22. Datasets, C.S.C.D.-. Water Meter Data.
23. Shvachko, K., et al. The hadoop distributed file system. in 2010 IEEE 26th symposium on mass storage systems and echnologies (MSST). 2010. Ieee.
24. GeeksforGeeks. Hadoop - Pros and Cons. Available from: https://www.geeksforgeeks.org/hadoop-pros-and-cons/.
25. Ono, K., et al., HIVE: A cross-platform, modular visualization framework for large-scale data sets. Future Generation Computer Systems, 2020. 112: p. 875-883.
26. Vera-Baquero, A., R. Colomo-Palacios, and O. Molloy, Measuring and querying process performance in supply chains: an approach for mining big-data cloud storages. Procedia Computer Science, 2015. 64: p. 1026-1034.
27. Yaqoob, I., et al., WITHDRAWN: Information fusion in social big data: Foundations, state-of-the-art, applications, challenges, and future research directions. 2016, Elsevier.
28. Suthaharan, S., Machine learning models and algorithms for big data classification. Integr. Ser. Inf. Syst, 2016. 36: p. 1-

29. Kambatla, K., et al., Trends in big data analytics. Journal of parallel and distributed computing, 2014. 74(7): p. 2561-2573.
30. Dittrich, J., et al., Only aggressive elephants are fast elephants. arXiv preprint arXiv:1208.0287, 2012.
31. Chang, V., Towards a big data system disaster recovery in a private cloud. Ad Hoc Networks, 2015. 35: p. 65-82.
32. Zhao, Junlei, Chunxiao Li, and Lei Wang. "Hadoop-based power grid data quality verification and monitoring method." Journal of Electrical Engineering & Technology 18, no. 1 (2023): 89-97.
33. Aqsa Ijaz, Ammar Ahmad Khan, Muhammad Arslan, Ashir Tanzil, Alina Javed, Muhammad Asad Ullah Khalid, & Shouzab Khan. (2024). Innovative Machine Learning Techniques for Malware Detection. Journal of Computing & Biomedical Informatics, 7(01), 403–424.
34. Ammar Ahmad Khan , Muhammad Arslan , Ashir Tanzil , Rizwan Abid Bhatty , Muhammad Asad Ullah Khalid , Ali Haider Khan. (2024). Classification Of Colon Cancer Using Deep Learning Techniques On Histopathological Images. Migration Letters, 21(S11), 449–463
35. Hussain, S.K., Ramay, S.A., Shaheer, H., Abbas T., Mushtaq M.A., Paracha, S., & Saeed, N. (2024). Automated Classification of Ophthalmic Disorders Using Color Fundus Images, Volume: 12, No: 4, pp. 1344-1348 DOI:10.53555/ks.v12i4.3153
36. Abbas, F., Iftikhar, A., Riaz, A., Humayon, M., & Khan, M. F. (2024). Use of Big Data in IoT-Enabled Robotics Manufacturing for Process Optimization. Journal of Computing & Biomedical Informatics, 7(01), 239-248.
37. Munir, A., Sumra, I. A., Naveed, R., & Javed, M. A. (2024). Techniques for Authentication and Defense Strategies to Mitigate IoT Security Risks. Journal of Computing & Biomedical Informatics, 7(01).
38. Khan, A. H., Malik, H., Khalil, W., Hussain, S. K., Anees, T., & Hussain, M. (2023). Spatial Correlation Module for Classification of Multi-Label Ocular Diseases Using Color Fundus Images. Computers, Materials & Continua, 76(1).