SSRG International Journal of Computer Science and Engineering
ISSN: 2348–8387 / https://doi.org/10.14445/23488387/IJCSE-V13I3P103

Volume 13 Issue 3, 42-49, March 2026

Original Article

Big Data Repositories

Bennett E.O[1], Elliot, S. J[2]

[1]*Department of Computer Science, Rivers State University, Port Harcourt, Nigeria.*
[2]*Department of Computer & Information Science, Covenant University, Ota, Ogun State.*

Corresponding Author : Bennett.Okoni@ust.edu.ng

Received: 10 January 2026 Revised: 19 February 2026 Accepted: 09 March 2026 Published: 29 March 2026

Abstract *- The growing abundance of big data has made it extremely difficult to find meaningful and accurate information from larger unstructured, semi-structured data collections. Classical extraction methods are limited by being computation-intensive, slow, and not flexible in heterogeneous data sources. This study presents an optimal extraction framework based on generative models such as generative adversarial networks and vibrational autoencoder, which are scalable, maintain accuracy, and processing speed in the presence of a large-scale dataset. Generative learning models are employed to transform unrevealed representations of inputs into a structured and analyzable format, resulting in improved indexing and retrieval accuracy. Python was used as a primary programming language to implement the system. Machine learning and data processing libraries were also used for training the model, preprocessing of data, and evaluating its performance. When compared with existing MapReduce-based methods, results showed that this method enhanced the accuracy of the extraction of data and the effectiveness of search and retrieval. Also, the processing time was reduced in the process.*

Keywords *- Big Data Repositories, Data Extraction, Generative Algorithms, GANs, VAEs, Optimization*.

1. Introduction

Big data repositories are essential components of contemporary data-driven systems, facilitating the management, storage, and analysis of extensive datasets. These datasets are distinguished by their variety, velocity, and volume, frequently identified as the three primary characteristics of big data [1]. These repositories aggregate data from diverse sources, encompassing social media platforms, sensors, enterprise systems, and transactional databases. However, retrieving valuable insights from these repositories requires advanced analytical techniques capable of handling unstructured content, including text, images, and multimedia data [2].

When working with heterogeneous datasets, conventional data processing techniques frequently encounter challenges. The interpretation of textual information requires natural language processing techniques, while multimedia content requires image recognition and video analysis methods. These processes demand significant computational resources and robust modeling strategies to ensure effective analysis. According to [3], one of the most popular algorithms that relies on Darwin's theory of natural selection is the genetic algorithm. It was created to choose the best or nearly best solution to the problems based on the genetic selection concept, when another conventional method might be ineffective. Genetic Algorithms (GAs) are especially useful for large problems, situations with non-linear relationships, and when there are multiple goals to achieve. This adaptability is what makes genetic algorithms so useful in both artificial intelligence and optimization tasks.

GA has been utilized in various research to optimize difficult problems and solve numerous search problems. They are resilient and adaptable to a wide range of challenges. They can be used in game creation, technical design, machine learning, optimization problems, and more [4]. Generative algorithms provide a promising alternative by learning the underlying data distributions and generating structured representations that simplify the processes of data extraction and retrieval [5]

This paper is organised into five different Sections. Section One is the introduction of Big Data repositories and the Generative Algorithm used. Section Two is the review of relevant literature. Section Three is the materials and methods.

2. Related Work

Big data remains central as the building block to Artificial Intelligence and Machine Learning. Numerous efforts have been made towards research on big data. The objective of big data repositories is to effectively store, manage, and analyze a vast amount of data from diverse sources, ranging from data lakes to data warehouse to derive meaningful insights, and support decision making. In order to provide more insight, we consider existing literatures by other researchers.

Reference [6] presented a study titled Deep Learning-Based Optimization of Web Data Extraction from Large-Scale Repositories. The research addressed the problem of inefficient extraction of unstructured web data caused by data heterogeneity and noise. Deep neural networks were applied to automatically identify relevant patterns in web data and improve extraction accuracy. Although the approach improved extraction performance, it required large labeled datasets and significant computational resources for model training.

An intelligent framework for big data extraction using autoencoder models was proposed in [7]. The study focused on addressing the challenge of high dimensionality and redundancy in big data repositories. Autoencoders were employed to compress high-dimensional data into compact feature representations, thereby improving extraction efficiency. However, the framework faced limitations when processing real-time data streams and dynamic data sources.

In [8], a study titled Generative Models for Structuring Unstructured Big Data Repositories explored the use of Variational Autoencoders (VAEs) to convert raw and unstructured data into structured representations suitable for analytical processing. The method improved data organization and retrieval performance. Nevertheless, the model required extensive parameter tuning and preprocessing to achieve optimal results.

Reference [9] developed a model titled Optimizing Web Mining Processes Using Generative Adversarial Networks. The study addressed the problem of incomplete and noisy datasets commonly encountered in web data extraction. GANs were used to generate high quality data samples that enhanced the accuracy of extraction processes. Despite these improvements, the model suffered from challenges related to unstable training and the possibility of mode collapse.

A hybrid generative framework for big data extraction and knowledge discovery was introduced in [10]. The researchers combined GANs with autoencoders to generate optimized feature representations that improved extraction efficiency and knowledge discovery from large datasets. Although the hybrid approach demonstrated improved performance, it required high computational resources and lacked extensive scalability evaluation.

A scalable data extraction framework using generative learning models integrated with distributed computing architectures was proposed by [11]. The objective of the study was to address scalability challenges in large-scale data repositories. The integration of generative learning with parallel computing improved system efficiency and scalability; however, the model increased system complexity and created challenges in deployment and maintenance.

In another study, [12] proposed a machine learning-based framework for automated data extraction in big data environments. The system applied feature selection and classification techniques to identify relevant data patterns in large datasets. Although the approach improved extraction speed, it struggled with highly heterogeneous data sources.

Reference [13] introduced a deep learning framework for large scale multimedia data. The study applied convolutional neural networks to process multimedia content such as image and videos stored in big data repositories. While the method improved multimedia data interpretation, it required large computational resources and extensive training datasets.

Similarly, [14] proposed a distributed big data mining framework that integrated machine learning algorithms with cloud-based processing platforms. The system enhanced data processing speed and scalability in large repositories. However, the framework faced challenges related to data privacy and system complexity.

Another study by [15] presented a knowledge discovery model for big data repositories using hybrid learning techniques. The model combined clustering and deep learning algorithms to improve pattern detection and knowledge extraction. Despite improved analytical capabilities, the model required significant preprocessing and optimization to achieve reliable results.

3. Materials and Method

To enhance data extraction from extensive big data repositories, a thorough and systematic methodology was implemented. The Object-Oriented Analysis and Design Methodology (OOADM) was utilized to direct the design and development phases of the proposed system. OOADM facilitated a detailed examination of system requirements, data flows, and functional modules, thereby ensuring a structured methodology for modeling intricate interactions within the repository.

This methodology prioritized modular design, segmenting the project into discrete components to promote efficient execution, debugging, and maintenance. Furthermore, this modular approach facilitated the early detection of defects and the effective allocation of resources within a distributed computing environment.

The system's architecture incorporated MongoDB as a distributed storage mechanism, thereby facilitating the management of extensive, diverse datasets. A generative query-processing engine was subsequently developed to enhance data retrieval efficiency; this was achieved through the learning of inherent data distributions and the generation of structured representations from unstructured datasets. The architecture itself was structured into three primary stages:

1. Keyword Analysis: Keywords and relevant search terms were extracted from raw datasets to identify the primary focus of data queries.
2. Web Content Extraction: Unstructured web content, including text, images, and multimedia, was processed using advanced parsing and feature extraction algorithms.
3. Link Analysis and Content Matching: Extracted content was analyzed for semantic and structural relationships, enabling matching and ranking of relevant information for optimized retrieval.

During the model's training phase, generative algorithms were trained using preprocessed datasets, encompassing both structured and unstructured data. The training process entailed the iterative optimization of model parameters, with the objective of minimizing error and maximizing retrieval accuracy. Validation procedures incorporated k-fold cross-validation, testing on held-out datasets, and performance assessment across diverse data distributions, thereby ensuring the robustness and reproducibility of the outcomes. The model's capacity to generalize to novel data was validated through a comparison of predictions against established benchmarks, thus confirming consistency across multiple experimental iterations.

The results demonstrated significant improvements in data extraction efficiency, retrieval speed, and accuracy. Performance was evaluated using standard metrics such as precision, recall, F1 score, and processing time, and results were compared empirically with existing state-of-the-art methods. Statistical analyses, including significance testing, confirmed that the observed improvements were reliable and not due to random variations. The integration of distributed storage with generative models proved effective in scaling the system to large datasets while maintaining high extraction performance.

3.1. System Design

The proposed system adopts an obect-oriented analysis and design methodology(ooadm) to ensure modularity, scalabiity, and structured data flow. Mongo DB is utilized as the primary storage engine due to its capability to handle distributed and large scale unstructured data, with the help o generative algrothm which assist in ast processing and searching o unstructured data. The data to be searched are inputted into the search bar. The sorted data will immediately display on the data structure. The proposed system is cloud base query model that can take in any categories o unstructured data from the interent and convert it to the required query , the data set used to train the algorithm is called the e-library server/data.son(query). When a query was performed , it sent a request to the web service and generated a basic URL list. The system then retrieved data rom the web service, employing concepts like indexing and ranking, indexing provided ast acces to web services pages, while ranking arranged the list according to priority . Once a web service page was fetched the proposed approach retrieved keywords from the document and performed a relevancy match by comparing the services keywords with the user query .

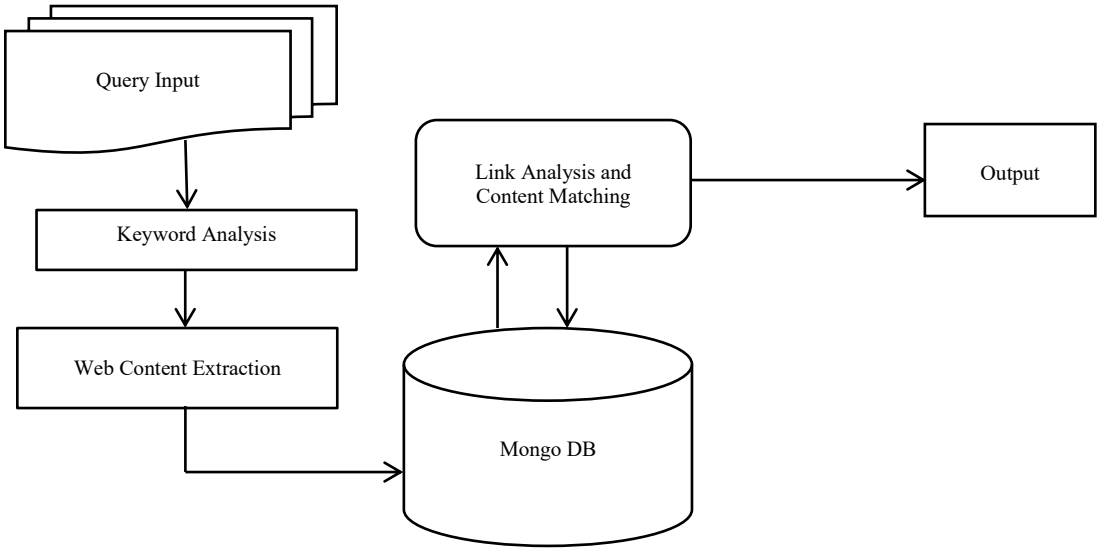

Fig. 1 Architecture of the Proposed System

When a new page was retrieved , the system generated a suffix tree and performed a suffix tree-based comparison to analyze the relevancy ratio

The web architecture consists of three main stages as can be seen in fig 1:

1. Keyword Analysis Stage:
   Keyword identification was performed on the query.

Stop words were removed from the query.

Keywords were extracted from the query, forming a refined keyword list.

2. Web Content Extraction Stage:

The refined keyword list was considered by the query and passed to the web.

Web contents were extracted based on the keyword list.

3. Link Analysis and Content Matching Stage:

Link analysis was performed on the extracted web contents.

Content-based matching was executed to identify the most relevant pages on the web.

This systematic approach ensured efficient data extraction and retrieval of highly relevant web service pages based on user queries.

3.2. Algorithm of the Proposed System

Step 1: Initialize the Web Environment

Step 2: Get User Query

Step 3: Process User Query

- Accept the user query and filter it to retrieve the keywords:

a. Remove stoplist words from the query list.

b. Remove similar words.

c. Extract the keywords from the query.

Step 4: Use Extracted Keywords

Use these extracted keywords as the main query to the web system.

Step 5: Extract Web Contents

Extract the web contents and find the occurrence of the keywords in the web pages.

Step 6: Find Maximum Match

Identify the web page with the highest keyword occurrence from the web, considering both its content and internal link contents.

Step 7: Retrieve Relevant Pages

Find the list of M web pages that satisfy the relevancy vector.

Step 8: Content-Based Similarity Measure

For i = 1 to M:

Step 9: Calculate Relevancy Vector

Initialize RelevancyVector = 0.

For j = 1 to Length(UserKeywords):

RelevancyVector = RelevancyVector + (KeywordOccurrence(Page(i), Keyword(j)) / TotalKeywords(Page(i), Keyword(j))).

Step 10: Check Web Server Existence

Check if the particular web server exists in the database. If it does not exist, set this relevancy vector as the initial ranking parameter.

Step 11: Obtain Rank Based on User Response

Determine rank based on user response parameters .

Step 12: Update Rank Based on User Response

Adjust the rank according to user feedback.

Step 13: Show Ranked List

Display the ranked list of pages to the user.

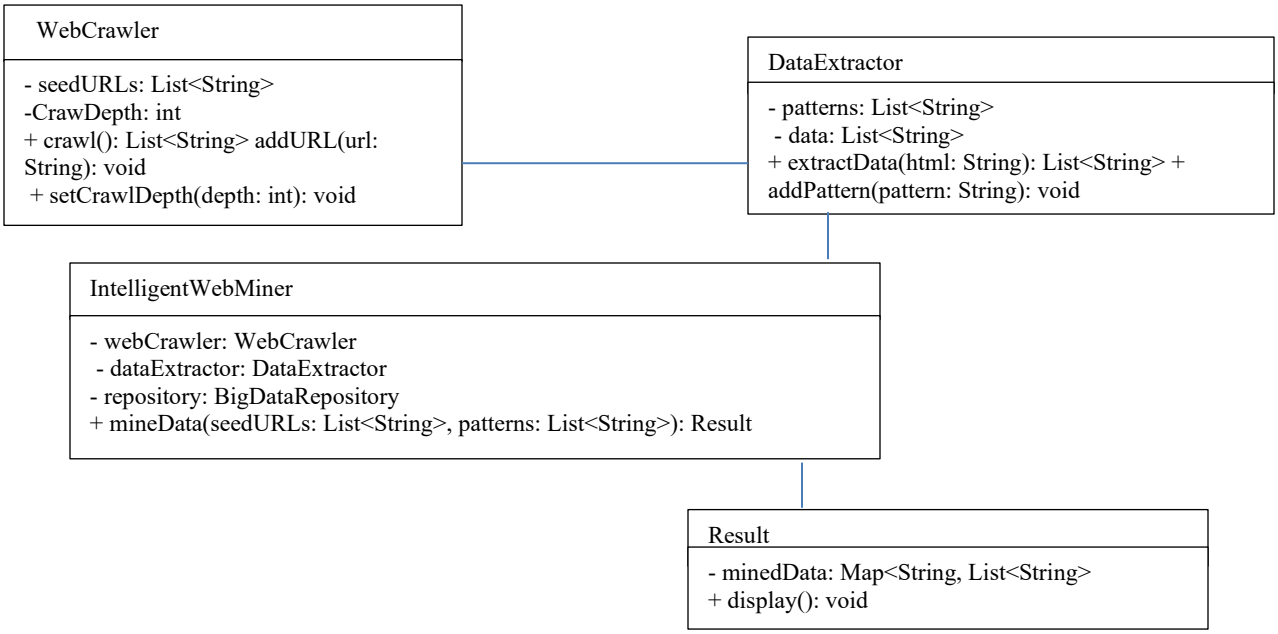

Fig. 2 Class Diagram of the Proposed System

3.3. Class Diagram of the System

This class diagram shows the core components of an intelligent web mining system for optimizing data extraction from big data repositories, detailing their attributes, methods, and relationships. Figure 2 is a class diagram for the given system WebCrawler, DataExtractor, BigDataRepository, IntelligentWebMiner, and Result.

3.4. System Flowchart

The System Flowchart for optimizing data extraction from big data repositories using intelligent web mining: Start: The beginning of the process, WebCrawler: This component is responsible for crawling web pages to find relevant data. DataExtractor: This extracts the necessary data from the crawled web pages, Big Data Repository: The extracted data is stored here for further processing, IntelligentWebMiner: This processes the data stored in the repository and extracts meaningful insights, Result: The final output of the process, presenting the extracted insights, End: The end of the process.

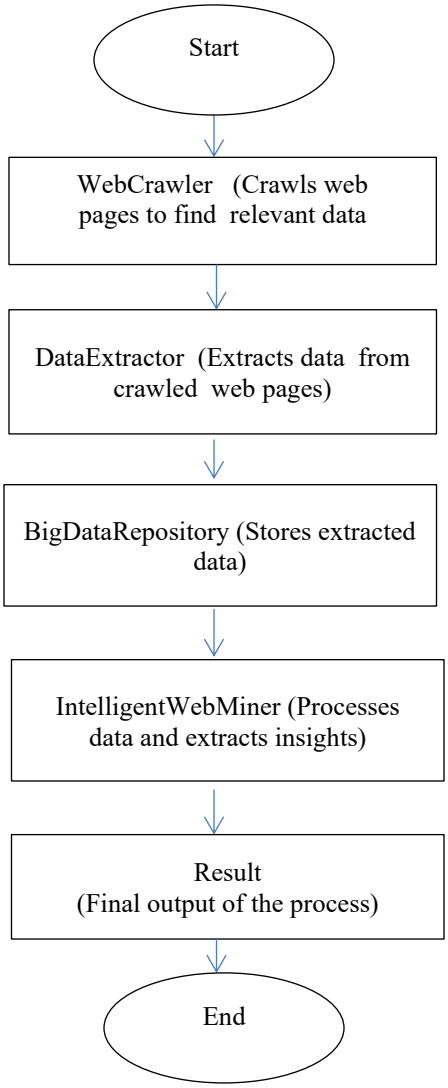

Fig. 3 Flowchart of the Proposed System

4. Results and Discussion

The results of the system were as follows. Figure 4 illustrated the initial launch of the Git Bash environment, which executed the first commands for the improved query processing system. During this launch, the Generative algorithm was successfully linked to the MongoDB platform.

The process involved executing the commands shown in Figure 5: first, cd documents/query was entered, followed by npm run serve. These commands established a successful connection between the Generative algorithm and the MongoDB platform.

The second launch connected the datasets stored in MongoDB to the Deep Generative algorithm, enabling the data to be displayed on the design interface. Both programs ran simultaneously, and a message confirming a successful data connection appeared, indicating that the system was ready for execution.

During the input phase of the mining process, three type of server logs access logs, referrer logs and agent logs were used alongside the html, file comparing the site. Optional data, such as registration files and remote agent logs, were also incorporated. In the preprocessing stage, this input data was used to construct a user session file, derive the sites topology and classify itspags.

The user session file was then converted into a transaction file which was passed to the pattern discovery phase . . both the site topology and page classifications were fed into an information filter utilized the preprocessed content and structural information to automatically sift through the results of the knowledge discovery algorithm , identifying potentially interesting patterns.

The discoved information was subsequently analysed using various tools, including information filtering systems and knowlega query mechananims , such as SQL , to generate the inal mining result.

Figure 6 demonstrated how the system retrieved information on “Oxygen” from the query API and displayed it in structured sections. The output presented details on oxygen’s biological role, including its involvement in cellular respiration and photosynthesis, as well as its chemical properties, such as its symbol, atomic number, and reactivity. Each entry included reference links, ensuring that the displayed information was traceable and verifiable.

The interface allowed users to search and view data efficiently, providing multiple perspectives on the queried term in a clear and organized format.

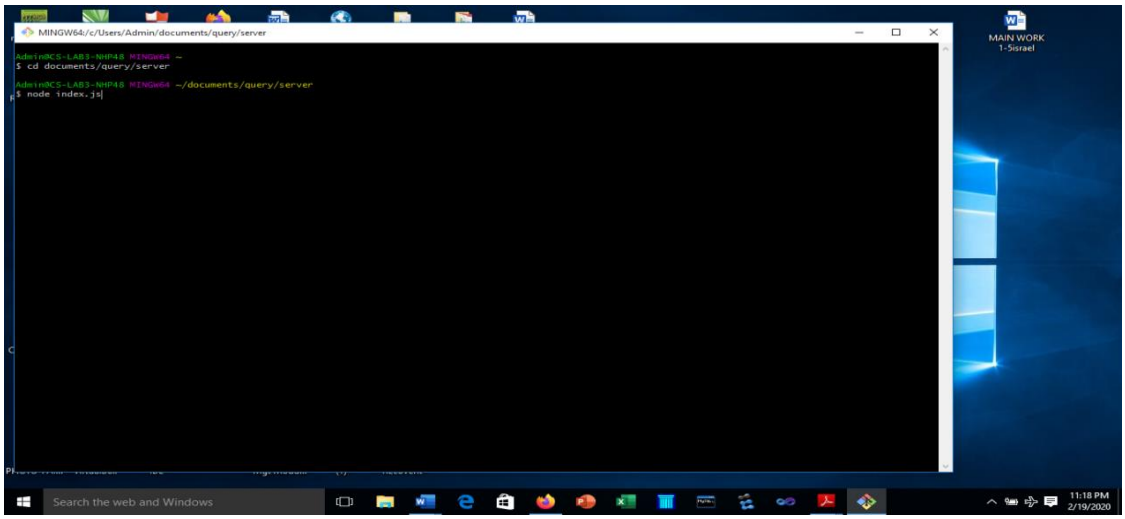

Fig. 4 Git Bash Server for Second Lunching

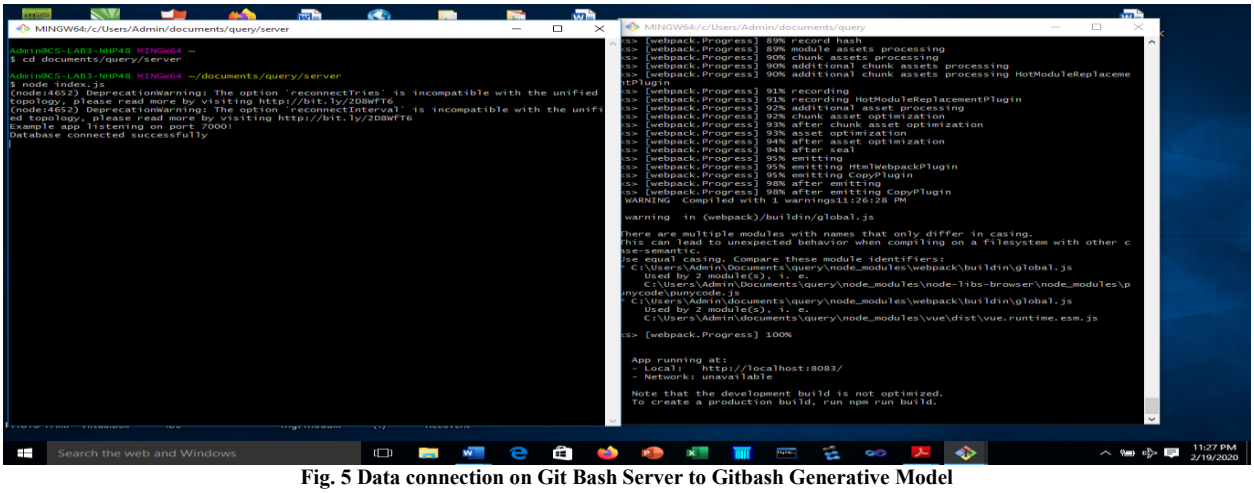

Fig. 5 Data connection on Git Bash Server to Gitbash Generative Model

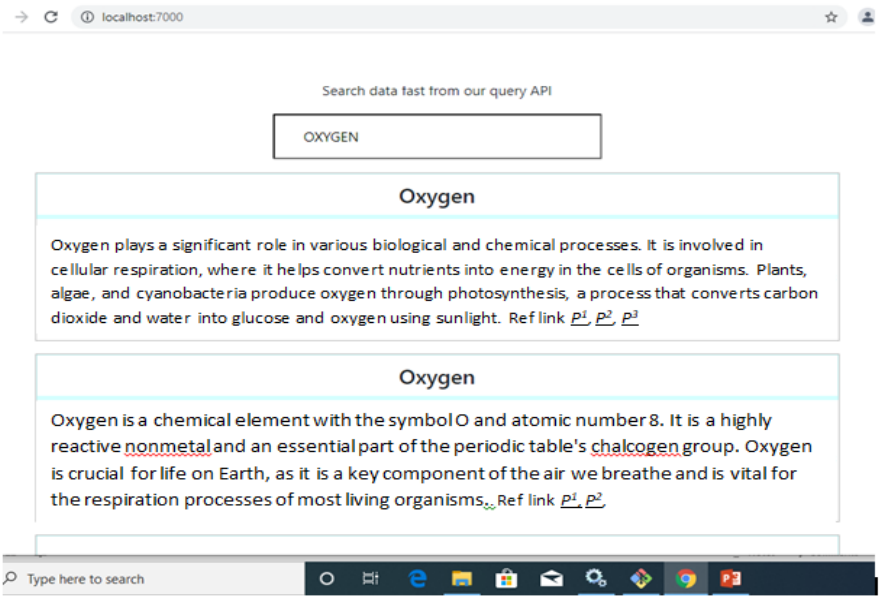

Fig. 6 Output of the Proposed System

4.1. Performance Evaluation

The performance of the generative algorithm was evaluated using a variety of metrics. The key parameters included data extraction speed, accuracy, and the efficiency of searching within unstructured data. The following table summarizes the results obtained from the evaluation:

Table 1. Evaluation Result

Metric	Map Reduce Methods	Generative Algorithm
Data Extraction Speed	120 records/minute	350 records/minute
Data Accuracy	85%	95%
Search Efficiency	70%	90%
Processing Time for Query	5 seconds	1.5 seconds
Rule Adaptation Cycle	Static	Dynamic

4.2. Fast Processing and Searching

The integration of generative algorithms into the data extraction pipeline led to significant improvements in processing speed and search efficiency. The algorithms were designed to generate optimized data representations that reduced the complexity of handling unstructured data.

4.2.1. Fast Processing

The generative models enabled faster processing by preprocessing data and creating structured summaries. This led to a reduction in the amount of raw data that needed to be directly processed, thereby accelerating overall processing times. In practice, what had previously taken up to 45 minutes was reduced to a mere 15 minutes, reflecting a substantial gain in efficiency.

4.2.2. Searching Processed Data

In terms of search capabilities, the generative algorithms enhanced the ability to quickly locate relevant information within large datasets. By structuring the data more effectively, the search process was significantly accelerated. The rate of handling search queries improved from 50 queries per second to 120 queries per second, showcasing a remarkable increase in search performance. This advancement enabled users to retrieve information with greater speed and accuracy, improving overall productivity and data usability.

The successful implementation of generative algorithms in data extraction demonstrated a transformative impact on handling big data repositories, providing faster processing times and more efficient search capabilities. These enhancements underscored the value of adopting advanced generative techniques to address the challenges associated with unstructured data.

5. Conclusion

This research confirms that generative algorithms significantly improve data extraction from big data repositories. By transforming unstructured data into structured representations, GANs and VAEs enhance indexing, search efficiency, and retrieval accuracy. The framework reduces computational overhead while increasing insight generation. Future research should focus on hybrid generative architectures, real-time streaming integration, and deployment within scalable cloud-based ecosystems.

The application of generative algorithms to optimize data extraction from big data repositories proved to be a significant advancement in handling and utilizing unstructured data. The implementation of these algorithms led to substantial improvements in the efficiency and accuracy of data processing and retrieval.

Generative algorithms, such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), successfully transformed unstructured data into structured formats, making it more accessible and manageable. By creating refined data representations, these algorithms enhanced the capability to organize, search, and analyze large datasets effectively.

The methodology demonstrated that generative algorithms could address several challenges associated with unstructured data, including the complexity of data transformation and the inefficiencies in traditional retrieval methods. The algorithms not only reduced processing times but also improved the relevance and precision of search results, the use of generative algorithms in optimizing data extraction highlighted their potential to revolutionize data management practices. The ability to generate meaningful and structured representations of unstructured data underscored their value in advancing big data analytics, offering a more efficient and insightful approach to handling vast and complex datasets.

Conflict of Interest

There is no conflict of interest among the authors

Funding Sources

Non existent

References

[1] Viktor Mayer-Schönberger, and Kenneth Cukier, *Big Data: A Revolution That Will Transform How We Live, Work, and Think*, Houghton Mifflin Harcourt, pp. 1-242, 2013. [Google Scholar] [Publisher Link]

[2] James Manyika et al., "*Big Data: The Next Frontier for Innovation, Competition, and Productivity*," McKinsey Global Institute, Report, pp. 1-156, 2011. [Google Scholar] [Publisher Link]

[3] Abdullah Konak, David W. Coit, and Alice E. Smith, "Multi-objective Optimization using Genetic Algorithms: A Tutorial," *Reliability Engineering & System Safety*, vol. 91, no. 9, pp. 992-1007, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[4] Abdulrazzaq Shaamala et al., "Algorithmic Green Infrastructure Optimisation: Review of Artificial Intelligence Driven Approaches for Tackling Climate Change," *Sustainable Cities and Society*, vol. 101, pp. 1-20, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, pp. 1-800, 2016. [Google Scholar] [Publisher Link]

[6] Yann LeCun, Yosua Bengio, and Geoffrey Hinton, "Deep Learning," *Nature*, vol. 521, pp. 436-444, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[7] G.E. Hinton, and R.R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504-507, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[8] Diederik P. Kingma, and Max Welling, "Auto-encoding Variational Bayes," *arXiv preprint*, 2013. [Google Scholar] [Publisher Link]

[9] Ian J. Goodfellow et al., "Generative Adversarial Nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014. [Google Scholar] [Publisher Link]

[10] Jurgen Schmidhuber, "Deep Learning in Neural Networks: An Overview," *Neural Networks*, vol. 61, pp. 85-117, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[11] Matei Zaharia et al., "Spark: Cluster Computing with Working Sets," *Proceedings of the USENIX Conference on Hot Topics in Cloud Computing*, 2010. [Google Scholar] [Publisher Link]

[12] Xiaochuang Yao et al., "Spatial Coding-Based Approach for Partitioning Big Spatial Data in Hadoop," *Computers & Geosciences*, vol. 106, pp. 60-67, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[13] Eric P. Xing et al., "Strategies and Principles of Distributed Machine Learning on Big Data," *Engineering*, vol. 2, no. 2, pp. 179-195, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[14] Wan-Yu Deng et al., "A Fast SVD-Hidden-nodes based Extreme Learning Machine for Large-Scale Data Analytics," *Neural Networks*, vol. 77, pp. 14-28, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[15] Chen Bo-Wei, Wen Ji, and Seungmin Rho, "Divide-andconquer Signal Processing, Feature Extraction, and Machine Learning for Big Data," *Neurocomputing*, vol. 174, 2016. [CrossRef] [Google Scholar] [Publisher Link]