Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 2860735, 11 pages
https://doi.org/10.1155/2022/2860735

Research Article

An Innovative Model for Extracting OLAP Cubes from NOSQL Database Based on Scalable Naïve Bayes Classifier

Farnaz Davardoost, Amin Babazadeh Sangar, and Kambiz Majidzadeh

Department of IT and Computer Engineering, Urmia Branch, Islamic Azad University, Urmia 57169-63896, Iran

Correspondence should be addressed to Amin Babazadeh Sangar; bsamin2@liveutm.onmicrosoft.com

Received 26 January 2022; Revised 15 March 2022; Accepted 25 March 2022; Published 11 April 2022

Academic Editor: Yiwen Zhang

Copyright © 2022 Farnaz Davardoost et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Due to unstructured and large amounts of data, relational databases are no longer suitable for data management. As a result, new databases known as NOSQL have been introduced. The issue is that such a database is difficult to analyze. Online analytical processing (OLAP) is the foundational technology for data analysis in business intelligence. Because these technologies were designed primarily for relational database systems, performing OLAP in NOSQL is difficult. We present a model for extracting OLAP cubes from a document-oriented NOSQL database in this article. A scalable Naïve Bayes classifier method was used for this purpose. The proposed solution is divided into three stages of preparation, Naïve Bayes, and NBMR. Our proposed algorithm, NBMR, is based on the Naïve Bayes classifier (NBC) and the MapReduce (MR) programming model. Each NOSQL database document with nearly the same attribute will belong to the same class, and as a result, OLAP cubes can be used to perform data analysis. Because the proposed model allows for distributed and parallel Naïve Bayes Classifier computing, it is appropriate and suitable for large-scale data sets. Our proposed model is a proper and efficient approach when considering the speed and reduced the number of required comparisons.

1. Introduction

Data generation has increased dramatically in recent decades. The rise of large-scale web platforms, such as Google, Facebook, Twitter, and Amazon, has resulted in the development of Big Data management approaches. The term "Big Data" refers to a massive amount of information. There are numerous theoretical definitions of Big Data [1]. The 3 V, Volume, Variety, and Velocity, is one of the most widely accepted definitions of Big Data [2]. The volume represents the massive amount of data in Petabytes and Exabytes. Data produced by various sources, such as smart devices and social media, results in structured, semistructured, and unstructured data. The velocity is related to the rate at which data is processed [3, 4].

Managing and analyzing Big Data is difficult. Because relational databases (RDBMS) are not suitable for large amounts of data, there is a growing interest in using NOSQL (Not Only SQL) database systems. NOSQL systems are an interesting alternative to relational databases because they are more scalable and flexible. The data structure of schema-oriented and highly structured relational database management systems should be known ahead of time. A relational database stores all data in tables as rows and columns, whereas a NOSQL database does not. Relational databases are an expensive solution for handling large amounts of data. Meanwhile, the NOSQL database scale-up approach is much simpler [5, 6].

The ACID principles are guaranteed by all relational databases. ACID (Atomicity, Consistency, Isolation, and Durability) is a set of database transaction properties designed to ensure that the database remains consistent in the event of a failure while processing a transaction. For this purpose, each transaction, which consists of operations, acts as a single unit, produces consistent results, is unaffected by any other transaction, and once committed, remains in the system.

Scalability and efficiency are achieved in NOSQL databases when ACID features are not strictly guaranteed, as opposed to relational databases, and BASE (Basically

Available, Soft state, Eventual consistency) features are ensured instead [7]. Thus, data repetition, sharing, and distribution on various storage services are used to increase system accessibility in the event of an event. In contrast to relational systems, which are strict about ensuring consistency, NOSQL systems allow data to be inconsistent in some cases. The system state may change as new data are added; NOSQL eventually ensures data consistency [8, 9].

Despite all of the benefits of NOSQL databases, analyzing such databases is difficult. Data should be analyzed for decision making in many complex contexts, including healthcare, security, and business. As a result, a suitable method for analyzing the NOSQL database should be developed. OLAP (online analytical processing) is a technique for processing and analyzing data. OLAP is a multidimensional structure that allows for quick access to data and advanced analysis [10, 11]. Because traditional OLAPs are based on relational databases, Big Data analysis on NOSQL databases is difficult [12].

One of the most common types of NOSQL databases is the document-oriented database. The document-oriented NOSQL database is the focus of this research. Data are stored in a collection of documents with different attributes in the document-oriented NOSQL database model [13, 14]. The primary goal of this research is to develop a model for converting a document-oriented NOSQL database to a structured model and extracting OLAP cubes. Previously, a variety of methods, such as rule-based, partitioning, similarity, and so on, were proposed. However, in previous solutions, Naïve Bayes was not used to extract the OLAP cube. There are two kinds of Naïve Bayes classifiers: traditional and parallel that is based on the Naïve Bayes classifier and the MapReduce programming model [15]. The Naïve Bayes classifier is a simple learning technique for text classification that uses the Bayes rule to map documents to the class with the highest probability calculated using Bayes' rules [16]. The most basic premise is that the attributes are conditionally independent of one another. When the data set is large, storage and processing can be difficult; to overcome these issues, a parallel MapReduce processing system with Naïve Bayesian is used [17, 18]. NBMR is the name of our proposed algorithm, which is scalable in large data sets. The scope of this study is depicted in Figure 1. The following is how the rest of the article is structured: The related work is presented in Section 2. The background of this study is presented in Section 3. The proposed model is presented in Section 4. The experimental results are examined in Section 5. Finally, Section 6 brings the article to a close.

2. Related Work

In this section, we look at the previous solutions that have been presented. Partitioning, chunking, rules (a set of rules for direct mapping from one model to another), parallel processing, MapReduce, and other methods have previously been proposed. Each solution has benefits and drawbacks.

The goal of Venkatesh and Ranjitha [19] was to build OLAP cubes out of a NOSQL database. To this end, the MC-CUB operator was created. This operator enables the

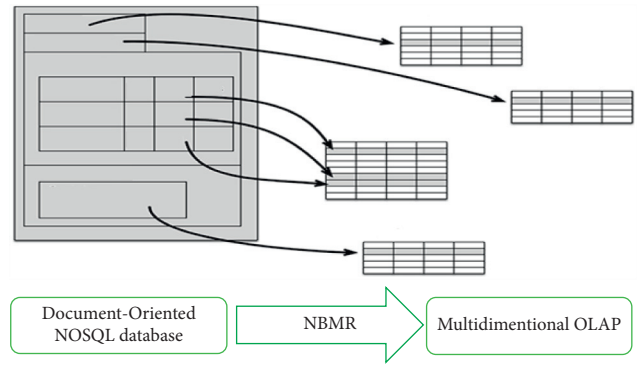

Figure 1: The scope of study.

processing of OLAP cubes on a large volume of data stored in a column-oriented NOSQL database. All possible aggregations have been performed at various levels and granularities. The MapReduce pattern was used to parallelize operations on stored data in a distributed environment. [20] Presents Halop, a solution for storing OLAP systems for Big Data, developed by the authors. To store data, chunking and partitioning methods are used in this solution. The data are also loaded using a parallel MapReduce processing system. NOSQL databases were used in decision-making systems in the study by Song et al. [21]. A set of rules has been defined to automatically and directly convert the multidimensional conceptual model to the logical NOSQL model. Data preaggregation has been considered in the proposed approach to improve query speed. This method has been investigated for both document-oriented and column-oriented NOSQL databases. They proposed a system for converting a NOSQL database to a relational database in [22]. Schema Analyzer first specifies the schema of the NOSQL database. Following the specification of the schema, ETL processes are started in parallel to extract, transform, and load data from a non-structured NOSQL database into a structured relational database. In the study by Wu et al. [23],they proposed a new model that combined the benefits of both relational and nonrelational databases in terms of logic and flexibility. They proposed a new soft set algebra called n-tier soft set for this purpose. This model allows data to describe itself and move and locate within a cluster with ease, but it also has the algebraic capabilities of a relational model, allowing for powerful queries. The data storage in the study by Chevalier et al. [24] was done with a document-oriented NOSQL database. The data model is specified with features and values in the proposed approach, and the data route is kept as a tree structure. To convert the multidimensional model to NOSQL, three map and conversion rules have been presented: All data cubes are stored in a set with a simple format in the first mapping rule (without subdocument). Subdocuments are stored in the set in the second rule. The cubes are kept in multiple distributed sets in the third rule, each with its own set of advantages and disadvantages. Chevalier et al. [25] investigated the dimensions with complicated hierarchy in two states: inaccurate (child level with multiple fathers) and no coverage (child level with no father), and a method for modelling the complicated

hierarchy to the document-oriented state was presented. Finally, the algorithm for aggregation is presented. Sohrabi and Azgomi [26] described an algorithm called MSJL for determining the similarity of large data sets. This algorithm is based on the LSH approximate similarity algorithm, which uses chunking and partitioning for Big Data management, as well as the MapReduce parallel processing system. Davardoost et al. [27] proposed a new method for extracting OLAP cubes from document-oriented NOSQL called LSHMR. The LSHMR programming model was based on the LSH and MapReduce programming models. The MapReduce framework enables distributed and parallel computing, while LSH is a fast and approximate similarity search that is used to reduce the number of comparisons. Vernica was created using the MapReduce programming model and prefix filtering. Similar sets must have a common token in their prefixes, according to the prefix filtering principle. It is a good filter that can cut down on the number of candidate pairs [28].

Various methods for converting an unstructured model to a structured model and vice versa have been presented. Table 1 summarizes the algorithms and approaches and compares them from various perspectives, including chunking and partitioning, using the MapReduce parallel processing technique, and rule-based processing.

3. Background

Because the goal of this article is to develop a model for extracting OLAP cubes from a document-oriented NOSQL database, the multidimensional conceptual model and the document-oriented NOSQL database are formalized first, followed by the MapReduce parallel programming model.

3.1. Formal Definition of the Multidimensional Conceptual Model. The multi-dimensional model includes fact tables and dimensions tables. The multidimensional scheme called E is defined as follows:

$$E = (F_E, D_E, \text{Snow Flake } E). \quad (1)$$

F_E is a limited set of the fact tables, N_F: name of the fact tables, and M_F: the measurement scale.

$F_E = \{F_1, \ldots, F_n\}$

$D_E = (N_F, M_F)$

$M_F = \{f1\ (M_1^F), f2\ (M_2^F), \ldots, f_v(m_v^F)\}$

$D_i = \{N_D, A_D, H_D\}$

(i) $\text{DE} = \{D_1, \ldots, D_M\}$ a limited set of dimensions

(ii) ND: name of dimensions of the multidimensional model

(iii) $\text{AD} = \{a_{1D}, \ldots, a_{UD}\}$

The set of attributes existing in the dimensions. There are two types of attributes, such that the simple attribute is an atomic and inseparable attribute and the complex attribute includes several attributes [29, 30].

(iv) $\text{HD} = \{H_{1D}, \ldots, H_{VD}\}$

HD: A set of hierarchies.

3.2. Formal Definition of the Document-Oriented NOSQL Database. NOSQL databases are seen as a viable alternative to relational databases, particularly in the context of Big Data. A schemaless database is a NOSQL database. NOSQL databases use documents instead of tables with data types, columns, rows, and schemas. Data are stored as a collection of documents in document-oriented databases [7, 31]. Each collection contains multiple documents, each of which has a dynamic structure. The collection is unstructured, and the documents that are already there have different fields [32]. The information is stored as pairs of keys and values in the structure of JSON or XML documents in the document-oriented NOSQL database [27, 33]. Figure 2 shows an instance of a document-oriented database. In document-oriented databases, the data are stored as a collection of documents as collection $= \{d_1, d_2, \ldots, d_n\}$. Each document in collection C has a unique key. The documents' structure is defined using attributes and values, where attributes are simple and composite: $d_i = \{(a_1, v_1), \ldots, (a_m, v_m)\}$ [31].

3.3. MapReduce. Figure 3 depicts a MapReduce overview. MapReduce is a popular parallel computing method for Big Data analysis. The MapReduce paradigm is straightforward, with two tasks: Map and Reduce. The input data set is divided into chunks by a Map, and each chunk is processed completely in parallel using Map tasks; the outputs of the Map phases are then sorted and used as the input of Reduce tasks [34].

Computation is executed as follows:

(i) Map tasks divide the data set into chunks and generate (Key-Value) pairs that are processed in parallel.

In the shuffle phase, the (Key-Value) pairs from each Map are collected and stored by their key. Then (Key-Value) pairs with the same key are passed to the same Reducer [35, 36].

(ii) Reduce task takes a new Key-Value pair (K, Value list) as input. The reducer combines all values with the same key [37].

4. Proposed Model

The goal of this article is to propose a model for OLAP cube extraction from a document-oriented NOSQL database. A scalable Naïve Bayes Classifier method was used for this purpose. The Scalable Naïve Bayes Classifier (SNBC) is a type of NBC that combines the Bayes method with the MapReduce programming model. NBMR is the name of our proposed solution. MapReduce is a popular parallel computing method for big data that allows NBC to scale up quickly, making this solution suitable for large amounts of data. The overview structure of our proposed model and its related phases is presented in this section. Figure 4 depicts the overall structure of our proposed model. Three phases of

Table 1: Comparison of the previous approaches.

Main idea	Chunk and partitioning	MapReduce	Similarity algorithm	Rule based	Classification
An approach called holap for storing OLAP system of big data	✓	MapReduce for loading data	✗	✗	✗
Presenting a set of rules for automatic conversion of the multidimensional model to the NOSQL model	✗	✗	✗	✓	✗
Constructing OLAP cube from column-oriented NOSQL database	✗	Using the MC-CUBE operator for MapReduce to parallelize the operations	✗	✗	✗
Presenting an approach for map OLAP to the document-oriented NOSQL database based on the tree structure	✗	✗	✗	✓	✗
MSJL algorithm for similarity of the collections	✓	✓	✓	✗	✗
Extracting OLAP cubes from document-oriented NOSQL database based on parallel similarity algorithms (LSHMR)	✓	✓	✓	✗	✗
Efficient parallel set-similarity joins using MapReduce (Vernica)	✓	✓	✓	✗	✗
Our proposed model	✗	✓	✗	✓	✓

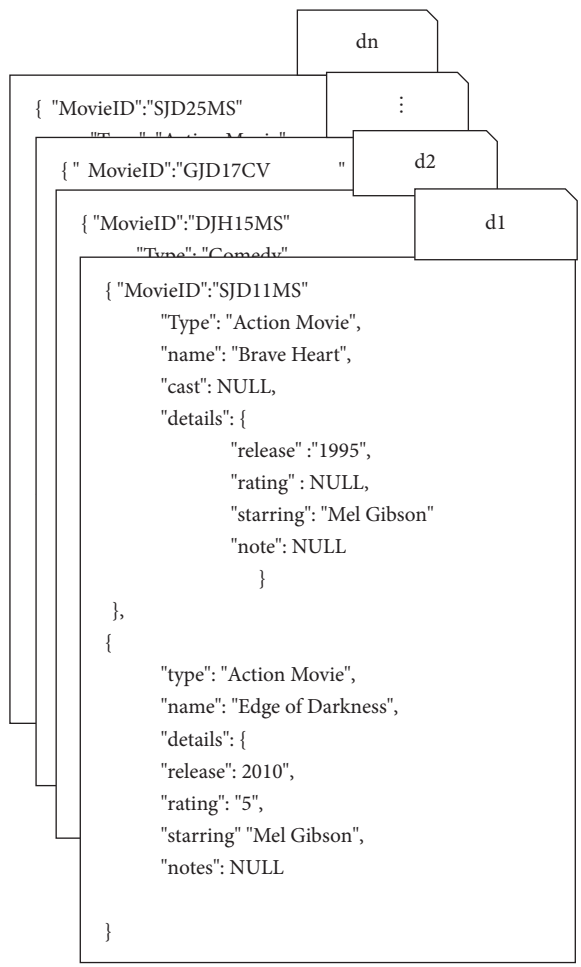

Figure 2: An example of document-oriented database.

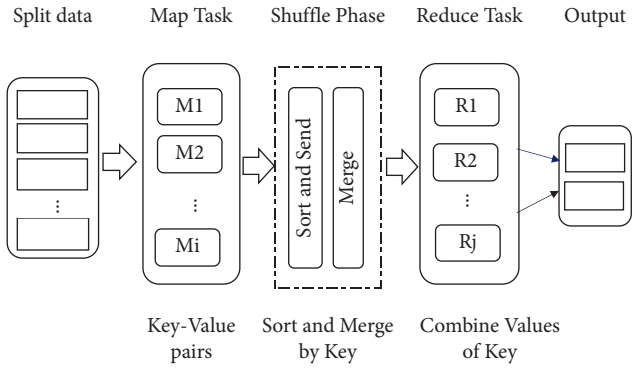

Figure 3: MapReduce programming model.

preparation, Naïve Bayes, and the NBMR algorithm are included in the proposed solution. The document-oriented NOSQL database is converted to a D matrix in the first phase, which is the preparation stage. The probabilities must be calculated using the Naïve Bayes theory in the second phase. The second phase is divided into three sections: Training, Combining, and Prediction. We use MapReduce in the third phase to parallelize and speed up the operation.

4.1. First Phase (Preparation Phase). The NOSQL document database is converted to a matrix in the first phase. The NOSQL document-oriented database is first used as input data in system. A document-oriented NOSQL database is made up of n documents that each have m attributes. Because it is unstructured, some attributes are present in each document, while others are absent. A D matrix with 0 and 1 elements is created. Matrix D's rows represent documents, columns represent attributes, and elements represent each document's attribute. If the attribute is null, the value is set to 0; otherwise, the value is set to 1. Matrix D is created based on these steps.

The second phase entails locating documents with similar attributes and classifying them accordingly. Because documents with similar structures belong to the same class, OLAP can be used to analyze data.

4.2. Second Phase (Naïve Bayes). In the second phase, the D matrix is subjected to the Naïve Bayes algorithm. A Naïve Bayes classifier is a learning algorithm that is based on the data set's conditional probability values. Every attribute is assumed to be independent of the others, which is a strong assumption. The classification principle of Naïve Bayes is relatively simple [38]. The probability of an element belonging to a class is calculated first, and then, the most likely element is chosen as the classification's result. The parts of Naïve Bayes are shown in detail in Figure 5. To begin, the matrix D is split into two sections: training and testing data. The probabilities are calculated using Bayes rules in the training phase, and a model is created. The test data is compared with the model in the combining section. If the model does not allow for test data, it is calculated and stored in an intermediate table. The prediction step is straightforward: use the model to determine which class the test data belong to [15, 39]. Similar documents in the structure are mapped to the same classes in the prediction section, and finally, OLAP cubes are extracted based on the classification.

The steps for classifying document-oriented NOSQL databases based on Naïve Bayes theory are as follows:

(1) Let Collection $= \{d_1, d_2, \ldots, d_n\}$

In document-oriented database, data are stored in a collection of documents. Each di represents a document that documents that are structurally similar will be in the same class.

(2) Let $A = \{a_1, a_2, \ldots a_m\}$

Each document has different attributes. Each a_i represents attributes of documents. It is assumed that all attributes of documents are m. Each document has some of these attributes while other attributes are null.

(3) Let $C = \{C_1, C_2, \ldots, C_N\}$

C represents different classes. We suppose there are N possible classes of training samples.

(4) Calculate probability of each document belongs to each class: $p(C_1|d), p(C_2|d), \ldots, p(C_N|d))$.

(5) If $p(C_k|d) = \max(p(C_1|d), p(C_2|d), \ldots, p(C_N|d))$, then the class of d is C_k.

In the following, the calculation of each of the conditional probabilities of step 4 is described. For example, conditional probabilities of the attributes in the documents are as follows: $p\ (a_1|C_1), P(a_2|C_1), \ldots, P(a_m|C_1)$; $P(a_1|C_2), P(a_2|C_2), \ldots p(a_m|C_N)$. Considering the attributes are conditionally independent, probability documents $p\ (C_i|d)$ based on the Naïve Bayes theory can be calculated according to equation (2).

$$P(C_i|d) = \frac{p(d|C_i).P(C_i)}{p(d)}. \tag{2}$$

Based on step 5, the class that generates the maximum value of equation (2) must be found. As can be seen from equation (2), because the denominator $P(d)$ is the same for all C_i, the maximum value of the numerator of equation (2) is considered. Considering the attributes are conditionally independent, equation (3) can be used.

$$\begin{aligned} P(d|C_i)P(C_i) &= P(a_1|C_i)P(a_2|C_i)\ldots P(a_m|C_i)P(C_i) \\ &= P(C_i)\prod_{j=1}^{m} P(a_j|C_i). \end{aligned} \tag{3}$$

4.3. Third Phase (NBMR). This section explains how to put our proposed model into action. Because calculating conditional probabilities in the third phase is time consuming, we use the MapReduce programming model to parallelize and speed up the process. Our proposed algorithm, dubbed NBMR, is a hybrid of Naïve Bayes classification (NBC) and the MapReduce distributed programming model. The Naïve Bayes classifier assigns a document to the class with the highest probability, as determined by Bayes' rule, and

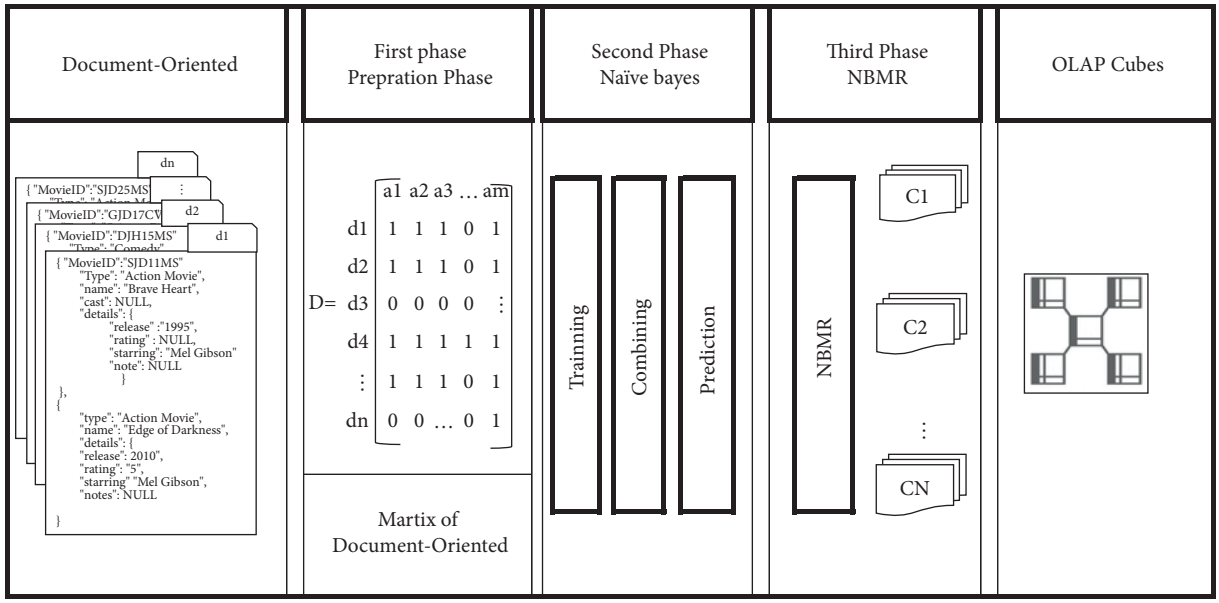

Figure 4: Overview of the proposed model.

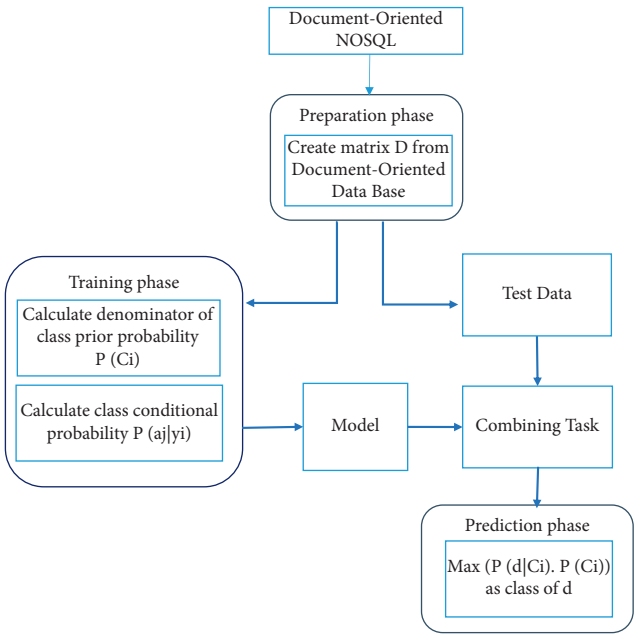

Figure 5: Second phase of proposed model.

MapReduce allows for parallel computing, which is ideal for large amounts of data. Master, Training, and Prediction were the three tasks we used to implement. The Master task controls the other threads as a server thread. Algorithms 1–3 shows the pseudocode for each task; each task performs the appropriate process on the input data. The input of the Master task is a document-oriented NOSQL database. The Preparation task is first called, and the document's matrix D is created. Matrix D is then divided into training and test data. The master task divides the training file "Tr" into k parts (Tr_1, Tr_2,..., Tr_k) and the testing file "T" into k parts (T_1, T_2,...,T_k), then calls the Training and Prediction task, which collects the results of these tasks and aggregates the probabilities that are used to classify the NOSQL database. Figure 6 depicts the process of putting our proposed model into action. Our proposed model, as shown in Figure 6, contains two sequential MapReduce tasks, the output of the first of which is the input of the second. The master task splits the training file "Tr" into K parts and assigns each part as a mapper input. Each mapper receives a portion of the data block and divides each record into specific Key-Value pairs Key = document, Value = attribute>, then computes the probability for each attribute based on each class. Each Key-Value is aggregated by class in the Reduce phase, and the model is created. The master task splits the testing file "T" into k sections. Each T part is assigned to mappers in the Prediction task. Each mapper receives a block of data from the previous MapReduce output during the Map phase. Using the model, each mapper classifies documents in T_i of the testing file. If the model does not have classification, classify by calculating probabilities and update the model. All results are aggregated in the Reduce phase, and eventually a class of documents is predicted, which is used to extract OLAP cubes. Table 2 contains a list of variables along with their definitions for quick reference.

5. Experimental Result

We run tests on a cluster with ten nodes, each with 12 GB of RAM and a 600 GB hard drive. For the NOSQL database, we used Hadoop 2.7.3 and the Mongo dB database. We compare our proposed solution to previous methods such as LSHMR [27] and Vernica to assess its performance. [28] The MapReduce programming model is used in both solutions. LSHMR is based on the nature of LSH and MapReduce, as well as Vernica, both of which are based on prefix filtering and the MapReduce programming model. We examine the runtime of our proposed solution using the DBLP database

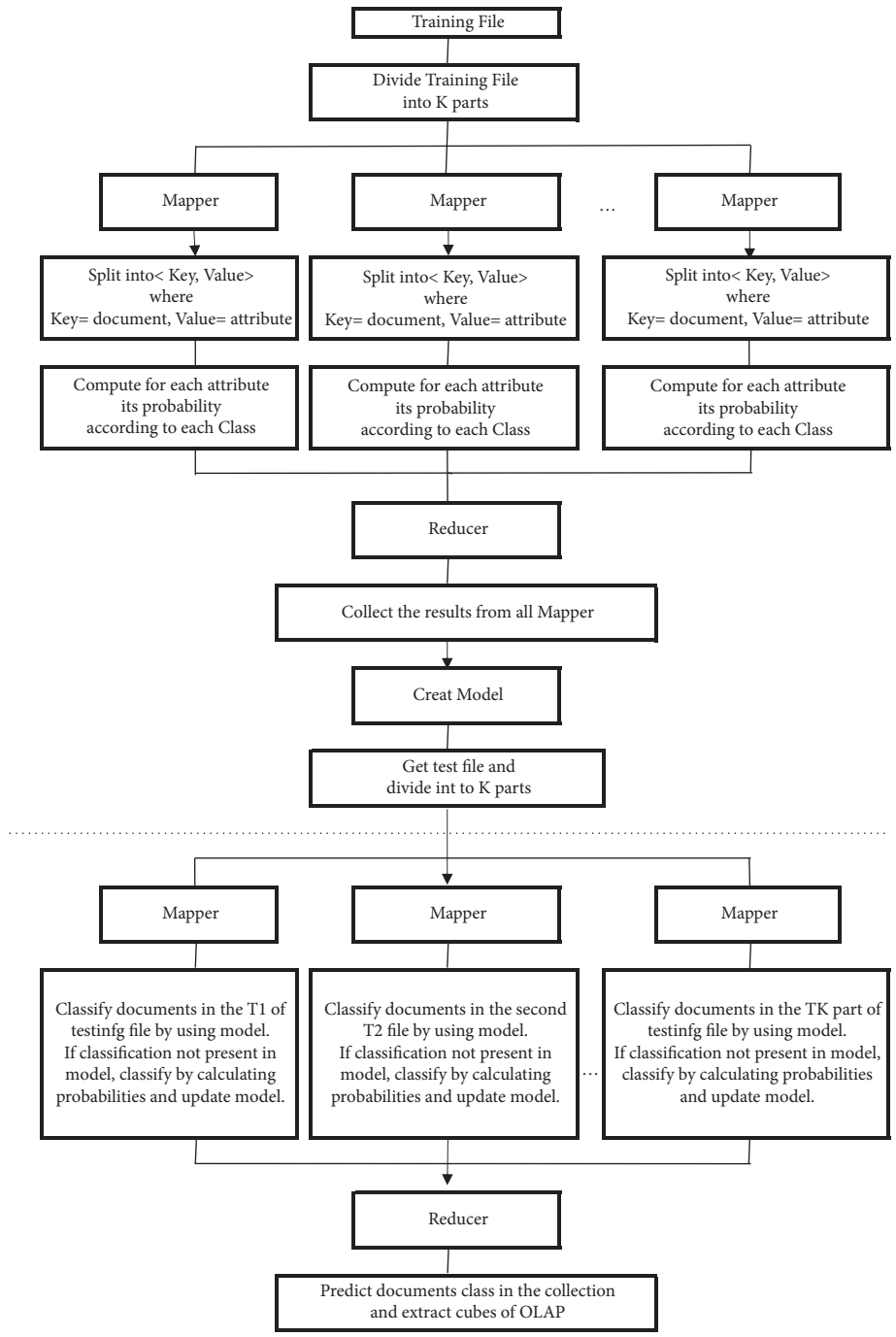

Figure 6: Our proposed solution (NBMR).

[40]. The number of nodes and the size of the database are used as evaluation metrics. The computer science bibliography database is known as DBLP. It has the largest open-access metadata collection of computer science journals and proceedings in the world. The XML format was used to save information, such as the author's name, article title, journal name, and so on. The runtime of each algorithm is depicted in Figure 7. On the DBLP database, we run these algorithms with various numbers of records ranging from 1000 to 50000. As shown in Figure 7, as the number of records grows, the runtime of algorithms grows, but the rate of time growth slows. Figure 7 shows that up to 10,000 records, all three algorithms take nearly the same amount of time to execute. The execution time of the algorithms changes as the number of records increases. All three algorithms have an increasing execution time trend in the 100,000 to 300,000 range, but NBMR performs better and takes less time than the other two algorithms in this range. The incremental slope of runtime in all three decreases as the number of records increases, and NBMR's performance is nearly identical to Vernica's. Several different nodes are used to evaluate the performance of NBMR in Figure 8. 2 nodes, 4 nodes, 6

Table 2: Description of variable.

Symbol	Description
D	Documents in a collection
N	Total number of documents
D	Matrix of collection
A	All attributes in the documents
A	Attributes of documents
M	Maximum number of documents attributes
C	Represent different classes
N	Total number of classes
$P(Cj)$	Possibility of jth class
$p(C_N\|d)$	Probability of each document belong to Nth class
$P(a\|C)$	Probability of attribute in class
Tr	Training file
T	Test file
K	Divide file in to K parts

Input: Document-oriented NOSQL database
Output: Classify documents of NOSQL data base and extract OLAP cube
Steps:
Call preparation task and create matrix D.
(1) Divide matrix D into Training and Test data.
(2) Divide Training file "Tr" into K parts,
(3) Tr_1, Tr_2,. . ., Tr_K.
(4) Calculate all probabilities and create model.
(5) Divide Testing file "T" into K parts,
(6) T_1,T_2,. . ..T_K.
(7) Send parts of file Test file, to prediction task.
(8) Use model and collect the classification result from
(9) Prediction task.
(10) Classify documents of NOSQL database and extract OLAP cubes.

Algorithm 1: Master task.

Input: Training file
Output: Model
Steps:
1: Assign each part of Tr_1,Tr_2, . . . Tr_k to mapper.
2: Split each document into a Key-Value pairs.
Where key = document and value = attribute
3: Compute all probabilities of $p(a_m|C_N)$ and $p(C_i)$.
4: Collect the result from all mapper.
5: Create model.

Algorithm 2: Training task.

Input: Testing file, Model
Output: Classify documents of NOSQL database and extract OLAP cubes
Steps:
1: Divide test file into T_n files and sent each file to mapper
2: For each documents in T_j file
IF (documents with same attributes

Algorithm 3: Continued.

```
    Has already been classified)
      Use the model
    Else
    For each attribute of document
      Calculate p(a_m|C_N)
End For
Classify document to class value with high probability
End IF
End For
```

Algorithm 3: Prediction task.

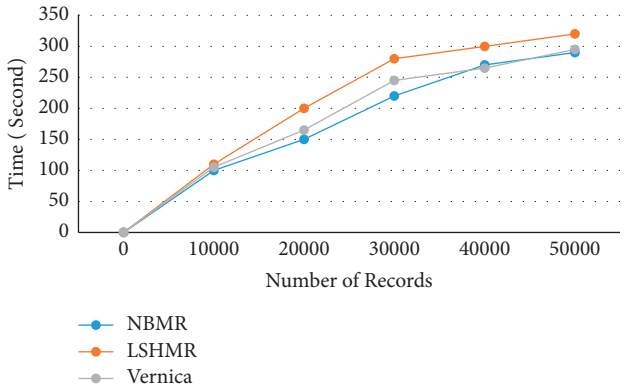

Figure 7: Performance of NBMR with different numbers of records.

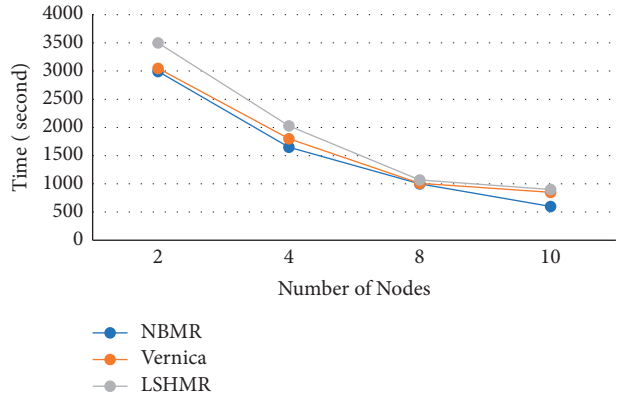

Figure 8: Performance of NBMR with different number of nodes.

nodes, and 10 nodes are used to test the run time of our proposed solution. NBMR and Vernica have similar performance at first, and the process is similar up to node 8, but as the number of nodes increases, NBMR outperforms the other two algorithms, and Vernica and LSHMR have similar performance.

6. Conclusions

Due to the diversity and large volume of data, the relational database is ineffective for data management. As a result, the use of NOSQL databases has grown in popularity recently. However, due to its unstructured nature, analyzing such a database is difficult. A new model for extracting data cubes is presented in this article. Partitioning, chunking, similarity, and rule-based solutions have all been proposed in previous studies. In this article, a new method for extracting OLAP cubes from document-oriented NOSQL databases is proposed. A scalable Naïve Bayes classifier method was used for this purpose. The proposed solution consists of three main preparation phases: Naïve Bayes, NBMR, and Naïve Bayes. The NOSQL document-oriented database is converted to a

matrix in the first phase. In the second phase, we use the Naïve Bayes theory to calculate probabilities. The second phase consists of three main tasks: training, combining, and prediction, all of which carry out a proper data process. We use MapReduce in the third phase to parallelize and speed up the operation. NBMR is the name of our proposed algorithm, which is based on the Naïve Bayes Classifier (NBC) and the MapReduce (MR) programming model. Each NOSQL database document with nearly the same attribute will be assigned to the same class, allowing OLAP cubes to be used for data analysis. The proposed model is appropriate and suitable for large-scale data sets because it allows distributed and parallel computing of the Naïve Bayes Classifier. Our proposed model is a proper and efficient approach, considering the speed and reduced number of required comparisons. Other machine learning techniques could be used in future research. We will concentrate on document-oriented NOSQL databases in this article; however, we can apply the same techniques to other types of NOSQL databases, such as graph oriented, column oriented, and so on.

Data Availability

The data used to support this study are obtained from [40].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] S. Sagiroglu and D. Sinanc, "Big data: a review," in *Proceedings of the 2013 International Conference on Collaboration Technologies and Systems (CTS)*, San Diego, CA, USA, May 2013.

[2] R. Hammad, M. Barhoush, and B. Abed-Alguni, "A Semantic-Based Approach for Managing Healthcare Big Data: A Survey," *Journal of Healthcare Engineering*, vol. 2020, Article ID 8865808, 12 pages, 2020.

[3] M. K. Pusala, M. Amini Salehi, J. R. Katukuri, Y. Xie, and V. Raghavan, "Massive data analysis: tasks, tools, applications, and challenges," in *Big Data Analytics: Methods and Applications*, S. Pyne, B. L. S. P. Rao, and S. B. Rao, Eds., Springer India, New Delhi, India, pp. 11–40, 2016.

[4] M. Bilal, L. O. Oyedele, J. Qadir et al., "Big Data in the construction industry: a review of present status, opportunities, and future trends," *Advanced Engineering Informatics*, vol. 30, no. 3, pp. 500–521, 2016.

[5] B. Jose and S. Abraham, "Performance analysis of NoSQL and relational databases with MongoDB and MySQL," *Materials Today Proceedings*, vol. 24, pp. 2036–2043, 2020.

[6] B. Namdeo and U. Suman, "Schema design advisor model for RDBMS to NoSQL database migration," *International Journal on Information Technology*, vol. 13, no. 1, pp. 277–286, 2021.

[7] G. Karnitis and G. Arnicans, "Migration of relational database to document-oriented database: structure denormalization and data transformation," in *Proceedings of the 2015 7th International Conference on Computational Intelligence, Communication Systems and Networks*, Riga, Latvia, June 2015.

[8] G. Deka, "BASE analysis of NoSQL database," *Future Generation Computer Systems*, vol. 52, 2015.

[9] H. Jing, E. Haihong, L. Guan, and D. Jian, "Survey on NoSQL Database," in *Proceedings of the 2011 6th International Conference on Pervasive Computing and Applications*, Port Elizabeth, October 2011.

[10] R. K. Lomotey and R. Deters, "Unstructured data, NoSQL, and terms analytics," in *Big Data Applications and Use Cases*, P. C. K. Hung, Ed., Springer International Publishing Cham, Manhattan, NY, USA, pp. 109–143, 2016.

[11] A. D. Patil and N. D. Gangadhar, "OLaaS: OLAP as a service," in *Proceedings of the 2016 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, Bangalore, India, October 2016.

[12] M. Bouakkaz, Y. Ouinten, S. Loudcher, and Y. Strekalova, "Textual aggregation approaches in OLAP context: a survey," *International Journal of Information Management*, vol. 37, no. 6, pp. 684–692, 2017.

[13] N. R. Gayathiri and A. M. Natarajan, "Big data retrieval using locality-sensitive hashing with document-based NoSQL database," *IETE Journal of Research*, vol. 67, no. 6, pp. 969–978, 2021.

[14] I. Ben Messaoud, A. A. Alshdadi, and J. Feki, "Building a document-oriented warehouse using NoSQL," *International Journal of Operations Research and Information Systems*, vol. 12, no. 2, pp. 33–54, 2021.

[15] P. Liu, H.-h. Zhao, J.-y. Teng, Y.-y. Yang, Y.-f. Liu, and Z.-w. Zhu, "Parallel naive Bayes algorithm for large-scale Chinese text classification based on spark," *Journal of Central South University*, vol. 26, no. 1, pp. 1–12, 2019.

[16] J. Chen, H. Huang, S. Tian, and Y. Qu, "Feature selection for text classification with Naïve Bayes," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5432–5435, 2009.

[17] M. M. Temesgen and D. T. Lemma, "A scalable text classification using naive Bayes with Hadoop framework," in *Information and Communication Technology for Development for Africa*Springer International Publishing Cham, Manhattan, NY, USA, 2019.

[18] B. Seref and E. Bostanci, "Sentiment analysis using naive Bayes and complement naive Bayes classifier algorithms on Hadoop framework," in *Proceedings of the 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Ankara, Turkey, October 2018.

[19] Venkatesh and K. V. Ranjitha, "Classification and optimization scheme for text data using machine learning Naïve Bayes classifier," in *Proceedings of the 2018 IEEE World Symposium on Communication Engineering (WSCE)*, Singapore, December 2018.

[20] K. Dehdouh, "Building OLAP cubes from columnar NoSQL data warehouses," in *Model and Data Engineering*Springer International Publishing Cham, Manhattan, NY, USA, 2016.

[21] J. Song, C. Guo, Z. Wang, Y. Zhang, G. Yu, and J.-M. Pierson, "HaoLap: a Hadoop based OLAP system for big data," *Journal of Systems and Software*, vol. 102, pp. 167–181, 2015.

[22] Z. Aftab, I. Waheed, A. Khaled Mohamad, B. Faisal, and A. Muhammad, "Automatic NoSQL to Relational Database Transformation with Dynamic Schema Mapping," *Scientific Programming*, vol. 2020, Article ID 8813350, 13 pages, 2020.

[23] J. Wu, D. Ni, and Z. J. M. I. S. Xiao, "N-tier Soft Set Data Model: An Approach to Combine the Logicality of SQL and the Flexibility of NoSQL," *Mobile Information Systems*, vol. 202123 pages, 2021.

[24] M. Chevalier, M. El Malki, A. Kopliku, O. Teste, and R. Tournier, "How can we implement a multidimensional data warehouse using NoSQL?" in *Enterprise Information*

*Systems*Springer International Publishing Cham, Manhattan, NY, USA, 2015.

[25] M. Chevalier, E. M. Mohammed, K. Arlind, T. Olivier, and T. Ronan, "Document-oriented Data Warehouses: Complex Hierarchies and Summarizability," in *Proceedings of the International Symposium on Ubiquitous Networking*, Springer, Singapore, May 2016.

[26] M. K. Sohrabi and H. Azgomi, "Parallel set similarity join on big data based on locality-sensitive hashing," *Science of Computer Programming*, vol. 145, pp. 1–12, 2017.

[27] F. Davardoost, A. Babazadeh Sangar, and K. Majidzadeh, "Extracting OLAP cubes from document-oriented NoSQL database based on parallel similarity algorithms," *Canadian Journal of Electrical and Computer Engineering*, vol. 43, no. 2, pp. 111–118, 2020.

[28] R. Vernica, M. J. Carey, and C. Li, "Efficient Parallel Set-Similarity Joins Using Mapreduce," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, Indianapolis, Indiana, USA, June 2010.

[29] A. Cuzzocrea, R. Moussa, and G. Xu, "OLAP∗: effectively and efficiently supporting parallel OLAP over big data," in *Model and Data Engineering*Springer, Berlin, Heidelberg, 2013.

[30] A. Cuzzocrea, L. Bellatreche, and I.-Y. Song, "Data warehousing and OLAP over big data: current challenges and future research directions," in *Proceedings of the Sixteenth International Workshop on Data Warehousing and OLAP*, San Francisco, California, USA, October 2013.

[31] M. Chavalier, E. M. Mohammed, A. Kopliku, and O. Teste, "Document-oriented data warehouses: models and extended cuboids, extended cuboids in oriented document," in *Proceedings of the 2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, Grenoble, France, June 2016.

[32] E. Gallinucci, M. Golfarelli, and S. Rizzi, "Schema profiling of document-oriented databases," *Information Systems*, vol. 75, pp. 13–25, 2018.

[33] B. Mustapha, L. Sabine, and O. Youcef, "Automatic textual aggregation approach of scientific articles in OLAP context," in *Proceedings of the 2014 10th International Conference on Innovations in Information Technology (IIT)*, Al Ain, UAE, November 2014.

[34] S. Bouaziz, A. Nabli, and F. Gargouri, "Design a data warehouse schema from document-oriented database," *Procedia Computer Science*, vol. 159, pp. 221–230, 2019.

[35] A. Fernández, S. d. Río, A. Bawakid, and F. Herrera, "Fuzzy rule based classification systems for big data with MapReduce: granularity analysis," *Advances in Data Analysis and Classification*, vol. 11, no. 4, pp. 711–730, 2017.

[36] Q. He, F. Zhuang, J. Li, and Z. Shi, "Parallel implementation of classification algorithms based on MapReduce," in *Rough Set and Knowledge Technology*Springer, Berlin, Heidelberg, 2010.

[37] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*, Cambridge University Press, Cambridge, UK, 2011.

[38] V. D. Katkar and S. V. Kulkarni, "A novel parallel implementation of Naive Bayesian classifier for Big Data," in *Proceedings of the 2013 International Conference on Green Computing, Communication and Conservation of Energy (ICGCE)*, Chennai, India, December 2013.

[39] H. Amazal, M. Ramdani, and M. Kissi, "A text classification approach using parallel naive Bayes in big data context," in *Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications*, Rabat, Morocco, October 2018.

[40] DPLB, "DBLP XML records," 2022, http://dblp.uni-trier.de/xml/.