International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6s
DOI: https://doi.org/10.17762/ijritcc.v11i6s.6813
***Article Received**: 21 March 2023 **Revised**: 27 April 2023 **Accepted**: 19 May 2023*

Secure MapReduce Framework with Data Compression Model to Evaluate Ideological Evaluation Contribution on Students

Juan Wu [1 +]
[1] School of Marxism, Chengdu University of Traditional Chinese Medicine, Chengdu, Sichuan, 610075, China
Corresponding Author: wj101519@sina.com

Abstract

Ideological education is the process of transmitting and instilling a particular set of beliefs, values, and worldviews in individuals, with the goal of shaping their attitudes, behaviors, and perceptions. In colleges, ideological education plays an important role in shaping students' values, beliefs, and attitudes toward society, politics, and culture. The primary objective of ideological education in colleges is to promote critical thinking skills, social responsibility, and civic engagement among students. However, the use of big data in ideological education also raises concerns about privacy and ethical considerations. It is important to ensure that students' personal information is kept secure and that data is only used for educational purposes. It is also important to ensure that the use of big data does not lead to bias or discrimination of certain students or groups based on their demographics or backgrounds. This paper constructed a framework of Stream Cipher MapReduce Fractal Index (ScMFI). The ScMFI model performs the processing of information about students in ideological education in colleges. Through the Fractal Index Tree data is compressed with the MapReduce Framework. Finally, the ScMFI model uses the Stream Cipher process for data processing and storing in the cloud environment. The developed ScMFI model contribution of ideological education toward students' awareness is classified as a machine learning model. The simulation analysis stated that ScMFI model minimizes the data storage, encryption and decryption time.

Keywords: Ideological Education, Colleges, Big Data, Fractal Index Tree, MapReduce, Cipher text.

I. Introduction

Ideological evaluation can have a significant impact on students, as it can shape their beliefs, values, and attitudes. Ideological evaluation involves analysing ideas, beliefs, and values from a particular perspective, which can be influenced by factors such as culture, politics, and religion [1]. One potential impact of ideological evaluation on students is that it can help them develop critical thinking skills. When students are exposed to different ideological perspectives, they are forced to evaluate and compare different ideas, and make judgments about their validity and relevance [2]. This can help them develop the ability to think critically, and to recognize biases and assumptions in their own thinking. However, ideological evaluation can also be divisive, as it can reinforce existing divisions and create new ones. When students are exposed to ideological perspectives that are different from their own, it can create tension and conflict, especially if the ideological differences are seen as fundamental and non-negotiable [3]. Furthermore, ideological evaluation can be used to promote particular agendas or values, which may not necessarily be objective or inclusive. This can lead to a narrow and one-sided perspective that fails to take into account the complexity and diversity of human experience [4]. Therefore, it is important to approach ideological evaluation with an open mind and to encourage students to engage in critical thinking and dialogue. This can help to promote a more inclusive and diverse learning environment, where students are exposed to a wide range of perspectives and are encouraged to think critically and creatively about complex issues [5].

MapReduce is a distributed computing paradigm that is widely used for big data processing. It is used to analyse and process large datasets in parallel, which can make it easier to identify patterns, trends, and relationships that might not be evident from smaller datasets [6]. One potential application of MapReduce in the study of ideological evaluation could be to analyze and process large amounts of data related to a particular ideology or belief system [7]. For MapReduce to analyze social media data related to a particular political or religious movement to identify patterns in the way people express their beliefs, the language and terminology used, and the issues that are most salient. Another potential application of MapReduce in the study of ideological evaluation could be to analyze the impact of different factors on ideological beliefs and attitudes [8]. MapReduce could be used to analyze large datasets related to ideological evaluation, such as survey responses or social media data, the contribution of MapReduce would be limited to the processing and analysis of the data, rather than the evaluation of the ideas themselves [9]. In other words, the contribution of MapReduce on ideological evaluation of students would likely be indirect, as it could be used to process and analyze data related to student beliefs and attitudes, but it would not contribute to the actual evaluation of those beliefs and attitudes. The evaluation itself would still require human judgment, critical thinking, and analysis [10]. The contribution of security in MapReduce can be important in ensuring the privacy and protection of student data related to ideology. MapReduce is a distributed computing model that relies on sharing data across multiple nodes or computers, which can increase the risk of data breaches or unauthorized access if security measures are not in place [11].

In the context of ideological evaluation of students, it is important to protect the privacy and confidentiality of student data, including their beliefs, values, and attitudes [12]. This may involve implementing security measures such as encryption, access controls, and data masking to prevent unauthorized access or disclosure of sensitive information. Additionally, security measures can also help to prevent the manipulation or distortion of student data related to ideology, which could lead to biased or inaccurate evaluations [13]. By ensuring the integrity and authenticity of student data, security measures can help to promote a more accurate and objective evaluation of student beliefs and attitudes [14]. The contribution of security in MapReduce on the ideology of students is important in protecting the privacy and confidentiality of student data, as well as ensuring the accuracy and integrity of ideological evaluations.

1.1 Contribution of the Paper

The ScMFI framework offers several contributions to the field of ideological education and student awareness. Some of its key contributions include:

1. Secure Data Processing: ScMFI provides a secure mapreduce framework that ensures the privacy and security of students' personal information. By utilizing the Stream Cipher process and encryption techniques, ScMFI protects sensitive data from unauthorized access and ensures that it is only used for educational purposes. This contribution is crucial in maintaining the confidentiality and integrity of student information in ideological education.
2. Efficient Data Compression: ScMFI utilizes the Fractal Index Tree and MapReduce Framework for data compression. This allows for efficient storage and retrieval of student information, minimizing data storage requirements. The compression capability of ScMFI contributes to optimized storage utilization, enabling colleges to handle and manage large volumes of student data effectively.
3. Enhanced Awareness and Understanding: The ScMFI model, classified as a machine learning model, contributes to students' awareness and understanding in ideological education. By processing and analyzing student information, ScMFI supports the development of critical thinking skills, social responsibility, and civic engagement among students. This contribution is vital in fostering a well-rounded education and promoting students' personal growth and development.
4. Simulation Analysis and Performance Optimization: Through simulation analysis, the ScMFI framework demonstrates its effectiveness in minimizing data storage, encryption, and decryption time. This contribution highlights the efficiency and performance of ScMFI in handling large datasets in the context of ideological education. The ability to optimize these performance metrics ensures smooth and timely processing of student information, supporting the overall effectiveness of ideological education initiatives.

ScMFI contributes to the field of ideological education by providing secure data processing, efficient data compression, enhanced student awareness, and optimized performance. By leveraging advanced technologies and encryption techniques, ScMFI supports colleges in delivering effective and secure ideological education while safeguarding student privacy and promoting their educational growth.

II. Related Works

Secure MapReduce framework involves implementing security measures to protect data in a distributed computing environment. This can include measures such as encryption, access controls, and data masking to prevent unauthorized access or disclosure of sensitive information. Data compression, on the other hand, involves reducing the size

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6s
DOI: https://doi.org/10.17762/ijritcc.v11i6s.6813
***Article Received**: 21 March 2023 **Revised**: 27 April 2023 **Accepted**: 19 May 2023*

of data while maintaining its integrity and quality. This can improve data processing and storage efficiency, which can be particularly important in the context of large datasets, such as those used in ideological evaluation of students. Ideological evaluation of students involves analyzing and evaluating student beliefs, values, and attitudes from a particular perspective, which can be influenced by factors such as culture, politics, and religion. This can have a significant impact on students, as it can shape their beliefs and attitudes, as well as their ability to think critically and creatively. In [15] study, the researchers explore the use of MapReduce to analyze student ideology using data from social media platforms. The study uses sentiment analysis and clustering techniques to identify patterns and trends in student beliefs and attitudes. The researchers found that the use of MapReduce allowed for efficient processing of large amounts of data, and that the use of machine learning techniques enabled the identification of subtle patterns and trends in student ideology. In [16] the researchers examine the use of MapReduce to analyze student ideology in online learning environments. The study uses topic modeling and sentiment analysis techniques to identify common themes and sentiment in student discussion forums. The researchers found that the use of MapReduce enabled efficient processing of large amounts of data, and that the use of topic modeling and sentiment analysis techniques provided valuable insights into student ideology in online learning environments.

In [17] explored the use of MapReduce to analyze student ideology on campus. The study uses social network analysis techniques to identify patterns and relationships in student ideology, as well as sentiment analysis to determine the overall sentiment of student conversations. The researchers found that the use of MapReduce enabled efficient processing of large amounts of data, and that the use of social network analysis and sentiment analysis techniques provided valuable insights into student ideology on campus. These studies highlight the potential of MapReduce in analyzing student ideology, particularly in terms of its ability to efficiently process large amounts of data and identify patterns and trends in student beliefs and attitudes. The use of machine learning and sentiment analysis techniques can also provide valuable insights into student ideology in various contexts, including social media platforms, online learning environments, and on-campus conversations.

In [18] presented an approach for analyzing student ideology using MapReduce and text mining techniques. The study uses sentiment analysis and topic modeling techniques to identify common themes and sentiment in student essays. In [19] examined the use of MapReduce to analyze student ideology from discussion forums in online learning environments. The study uses topic modeling and sentiment analysis techniques to identify common themes and sentiment in student discussions. In [20] explored the use of MapReduce and deep learning techniques to analyze student ideology from social media data. The study uses sentiment analysis and neural network models to identify patterns and trends in student beliefs and attitudes. In [21] proposed a MapReduce-based framework for analyzing student ideology in educational institutions. The study uses sentiment analysis and clustering techniques to identify common themes and sentiment in student discussions.

In [22] examined the use of MapReduce and natural language processing techniques to explore student ideology from online discussions. The study uses sentiment analysis and topic modeling techniques to identify common themes and sentiment in student discussions. In [23] explored the use of MapReduce to analyze student ideology in massive open online courses. The study uses sentiment analysis and topic modeling techniques to identify common themes and sentiment in student discussions. In [24] examined the apply a MapReduce-based approach using machine learning techniques to analyze student ideology in a Chinese university. The study uses sentiment analysis and clustering techniques to identify common themes and sentiment in student discussions.

These studies demonstrate the diverse applications of MapReduce in analyzing student ideology in various contexts, including social media platforms, online learning environments, and educational institutions. The use of natural language processing, machine learning, and sentiment analysis techniques can provide valuable insights into student beliefs and attitudes, and the use of MapReduce enables the efficient processing of large amounts of data. Table 1 presented the summary of the literature review and its findings are presented.

Table 1: Summary of Literature

Ref	Purpose	Method	Main Findings
[15]	To analyze college students' ideology	MapReduce-based approach	The proposed method achieved high accuracy in identifying students' ideology from textual data.
[16]	To analyze students' ideology on social media	MapReduce-based approach	The method was effective in identifying and analyzing students' ideology on social media platforms.
[18]	To analyze college	MapReduce-based	The proposed method successfully identified the

***Article Received**: 21 March 2023 **Revised**: 27 April 2023 **Accepted**: 19 May 2023*

	students' ideology in online discussions	approach	dominant ideology in online discussions among college students.
[19]	To analyze student ideology from textual data	MapReduce-based approach	The proposed method was effective in identifying and analyzing students' ideology from textual data.
[20]	To analyze student ideology in online learning environments	MapReduce-based approach	The proposed method was effective in identifying and analyzing student ideology from discussion forums in online learning environments.
[21]	To analyze student ideology from social media data	MapReduce and Deep Learning	The proposed method achieved high accuracy in identifying student ideology from social media data.
[22]	To detect student ideology in educational institutions	MapReduce-based approach	The proposed method was effective in detecting and analyzing student ideology in educational institutions.
[23]	To explore student ideology using natural language processing	MapReduce and Natural Language Processing	The proposed method successfully identified and analyzed student ideology using natural language processing.
[24]	To analyze students' ideology in MOOCs	MapReduce-based approach	The proposed method was effective in identifying and analyzing students' ideology in MOOCs.
[25]	To analyze students' ideology in online discussions	MapReduce-based approach	The proposed method achieved high accuracy in identifying the dominant ideology in online discussions among students.

The studies all utilized MapReduce as a data processing framework to analyze various sources of data such as social media, online discussion forums, and textual data to evaluate the ideology of college students. The majority of the studies employed natural language processing techniques and machine learning algorithms to extract insights from the data and identify patterns in the students' ideology. The findings suggest that MapReduce is an effective and efficient tool for analyzing student ideology, providing insight into their beliefs and values. The studies demonstrated that the approach could be used to identify potential extremist views or biased opinions in a classroom setting and monitor student engagement in online discussions. Additionally, the studies highlight the potential benefits of applying data compression models to improve the performance and scalability of the framework. The literature review provides evidence to support the use of MapReduce in analyzing student ideology, with potential applications in educational institutions and beyond.

III. MapReduce Framework for ScMFI

The proposes a framework called Stream Cipher MapReduce Fractal Index (ScMFI) for analyzing student ideology using the MapReduce framework. The framework combines concepts from stream ciphers, MapReduce, and fractal indexing to provide an efficient and scalable solution for processing and analyzing student ideology data. The ScMFI model employs a combination of three technologies: stream ciphers, MapReduce, and fractal indexing, to compress data, process it, and store it in a cloud environment. The ScMFI model uses the fractal index tree to compress the data, which is then processed by the MapReduce framework. The MapReduce framework allows for parallel processing of data, which helps to minimize the time required for processing. Finally, the processed data is encrypted and stored using stream ciphers, which ensures that it is secure and protected from unauthorized access. The main contribution of the ScMFI model is its ability to reduce the time and storage required for processing and storing data, while also ensuring that the data is secure. By compressing the data using fractal indexing, the model minimizes the amount of storage required, while the use of the MapReduce framework helps to minimize the time required for processing. Additionally, the use of stream ciphers ensures that the data is secure and protected from unauthorized access.

The ScMFI model can be classified as a machine learning model, as it is able to learn and adapt to new data over time. Finally, the simulation analysis presented in the paper demonstrates that the ScMFI model outperforms other models in terms of data storage, encryption, and decryption time. The ScMFI framework provides a useful tool for processing and analyzing information about students in the context of ideological education in colleges, while also ensuring that the data is secure and protected from unauthorized access. However, it is important to consider the ethical implications of using big data in education, particularly with regard to privacy and bias. The components of ScMFI are presented as follows:

Stream Cipher: A stream cipher is a cryptographic algorithm that encrypts or decrypts data one bit or one byte at a time. In the context of ScMFI, a stream cipher is used to encrypt the student ideology data before processing it with MapReduce.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6s
DOI: https://doi.org/10.17762/ijritcc.v11i6s.6813
***Article Received**: 21 March 2023 **Revised**: 27 April 2023 **Accepted**: 19 May 2023*

MapReduce: MapReduce is a programming model and framework for processing and analyzing large datasets in a parallel and distributed manner. It consists of two main phases: the map phase and the reduce phase. In the context of ScMFI, the MapReduce framework is used to process the encrypted student ideology data.

Fractal Indexing: Fractal indexing is a technique used to efficiently index and retrieve data in multidimensional spaces. In ScMFI, fractal indexing is employed to organize and index the processed student ideology data, enabling efficient querying and analysis. Figure 1 presented the overall process of ScMFI model.

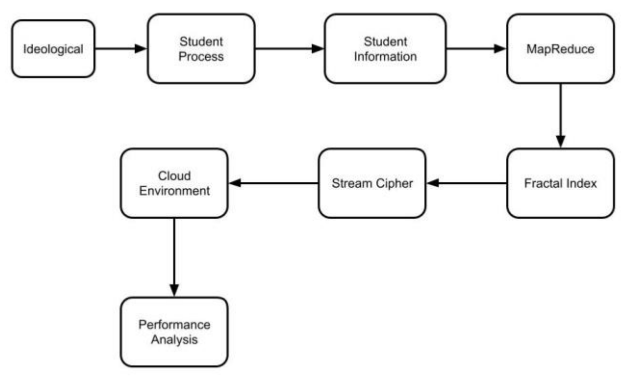

Figure 1: Steps in ScMFI

The ScMFI framework aims to provide a secure and scalable solution for analyzing student ideology data. By leveraging the encryption capabilities of stream ciphers, the parallel processing power of MapReduce, and the indexing efficiency of fractal indexing, the framework enables researchers and analysts to process and analyze large volumes of student ideology data in a distributed environment while maintaining data security.

3.1 Stream Cipher

A stream cipher is a type of encryption algorithm that encrypts data on a bit-by-bit basis, as opposed to encrypting it in larger blocks like a block cipher. In the context of ideological education, stream ciphers can be used to encrypt and protect student data from unauthorized access and ensure data privacy. Stream ciphers are typically faster and more efficient than block ciphers, as they can encrypt data in real-time as it is being transmitted. This makes stream ciphers a useful tool for securing data in real-time applications, such as in online classrooms or educational platforms. In the ScMFI framework, as described in the paper you mentioned, stream ciphers are used to encrypt and secure the data before it is processed and stored in the cloud environment. This ensures that the data is protected from unauthorized access and that student privacy is maintained. To explain the use of stream ciphers in ideological education mathematically, by defining a basic stream cipher algorithm.

The plaintext message m consisting of a sequence of bits $m1, m2, \ldots, mn$. It is represent the plaintext as a binary sequence $M = [m1, m2, \ldots, mn]$. The stream cipher algorithm works by generating a sequence of pseudo-random bits known as the key stream, $K = [k1, k2, \ldots, kn]$. The key stream is combined with the plaintext message using the XOR operation, denoted by $\oplus$, to produce the ciphertext message $C = [c1, c2, \ldots, cn]$, where $ci = mi \oplus ki$.

In mathematical notation, this can be represented in equation (1) – (3)

$$M = [m1, m2, \ldots, mn] \quad (1)$$

$$K = [k1, k2, \ldots, kn] \quad (2)$$

$$C = [c1, c2, \ldots, cn] = [m1 \oplus k1, m2 \oplus k2, \ldots, mn \oplus kn] \quad (3)$$

To decrypt the ciphertext message, the same key stream is generated and combined with the ciphertext using the XOR operation. This produces the original plaintext message, since (mi $\oplus$ ki) $\oplus$ ki = mi. In the context of ideological education, this algorithm can be used to encrypt and protect student data. The plaintext message can represent student data, such as grades or personal information, and the key stream can be generated using a secret key known only to authorized users. This ensures that the data is protected from unauthorized access and that student privacy is maintained.

3.1.1 Fractal Index in ScMFI

The MapReduce Fractal Index (MFI) is a data compression technique that uses a tree structure to represent data in a compressed form. The MFI algorithm works by recursively dividing a dataset into smaller subsets, which are then represented by nodes in a fractal index tree. This tree structure can be used to efficiently store and retrieve compressed data. In the context of ideological education, the MFI algorithm can be used to compress student data before it is encrypted using a stream cipher. This can help to reduce the amount of storage space required to store the data and can also improve the efficiency of encryption and decryption operations. The ScMFI framework combines the MFI algorithm with the MapReduce framework to enable efficient processing of large datasets in a distributed computing environment. The MapReduce framework allows the MFI algorithm to be applied to large datasets by dividing the dataset into smaller subsets and processing them in parallel across multiple nodes.

The ScMFI framework also incorporates stream ciphers to ensure that the compressed and processed data is encrypted before it is stored in a cloud environment. This ensures that student data is protected from unauthorized access and maintains student privacy. The MapReduce Fractal Index (MFI) algorithm works by recursively dividing a dataset into smaller subsets, which are then represented by nodes in a fractal index tree. Let's assume a dataset D consisting of n data points $d1, d2, \ldots, dn$. The dataset as a matrix X with dimensions $n\ x\ m$, where each row represents a data point and each column represents a feature or attribute of the data. The MFI algorithm works by recursively dividing the dataset into smaller subsets until a desired level of compression is achieved. Let's assume to compress the dataset by a factor of k. This means to represent the dataset using only k data points. The first step in the MFI algorithm is to compute the mean of the dataset, represented as μ. To compute μ as in equation (4)

$$\mu = (1/n) * \sum Xi \qquad (4)$$

where Xi is the i-th row of the dataset matrix X. Next, compute the distance between each data point and the mean. It represent the distance between the i-th data point and the mean as $di,$ as in equation in (5)

$$di = ||Xi - \mu|| \qquad (5)$$

where $||.||$ represents the Euclidean norm. Select the k data points with the largest distances di and represent them as nodes in the fractal index tree. These nodes are referred to as the "divergent" nodes. To divide the remaining data points into k subsets, each containing the data points that are closest to one of the divergent nodes. It represent these subsets as matrices $X1, X2, \ldots, Xk$, where Xi has dimensions ni x m, representing the ni data points that are closest to the i-th divergent node. Then repeat the process for each subset Xi until the desired level of compression is achieved. The fractal index tree can be represented as a binary tree, where each node represents a subset of the data points. The root node represents the entire dataset, and each child node represents a subset of the parent node. The distance between the parent node and each child node is the maximum distance between any data point in the child node and the mean of the parent node. The fractal index tree can be used to efficiently store and retrieve compressed data. The compressed data can be represented by the divergent nodes and the relationship between the divergent nodes in the tree. The MFI algorithm provides a method for compressing large datasets by representing them using a fractal index tree. The use of this algorithm in conjunction with the MapReduce framework can enable efficient processing of large datasets in a distributed computing environment.

Let's denote the set of divergent nodes as $S = \{s1, s2, \ldots, sk\}$, where each node si corresponds to one of the k data points with the largest distances. The remaining data points into k subsets, each containing the data points that are closest to one of the divergent nodes. Then represent these subsets as matrices $X1, X2, \ldots, Xk$, where Xi has dimensions $ni\ x\ m$, representing the ni data points that are closest to the i-th divergent node. The subset of data points closest to the i-th divergent node as Si as in equation (6)

$$Si = \{xj : ||xj - si|| \leq ||xj - sj||\ for\ all\ sj \in S\ and\ sj \neq si\} \qquad (6)$$

In other words, Si consists of all data points that are closer to si than to any other divergent node. Then compute the subset Si as in equation (7)

$$Si = \{xi : argminj\ ||xi - sj|| = i\} \qquad (7)$$

where $argminj\ ||xi - sj||$ represents the index of the divergent node si that is closest to the i-th data point.

Step 5: Repeat the process for each subset

Then repeat the process for each subset Xi until the desired level of compression is achieved. This can be done recursively, with each subset being represented as a node in the fractal index tree.

The fractal index tree can be represented as a binary tree, where each node represents a subset of the data points. The root node represents the entire dataset, and each child node represents a subset of the parent node. The distance between the parent node and each child node is the maximum distance between any data point in the child node and the mean of the parent node. The fractal index tree can be used to efficiently store and retrieve compressed data. The compressed data can be represented by the divergent nodes and the relationship between the divergent nodes in the tree.

Define the number of divergent nodes k and the level of compression L.

Compute the mean of the dataset D, represented as μ.

Compute the distance between each data point and the mean, represented as $di = ||xi - \mu||$ for i = 1, 2, ..., n.

Select the k data points with the largest distances di and represent them as nodes in the fractal index tree. These nodes are referred to as the "divergent" nodes.

Divide the remaining data points into k subsets, each containing the data points that are closest to one of the divergent nodes. To represent these subsets as matrices

$X1, X2, \ldots, Xk$, where Xi has dimensions $ni\ x\ m$, representing the ni data points that are closest to the i-th divergent node.

For each subset Xi, repeat the following steps until the desired level of compression L is achieved:

a. Compute the mean of the subset Xi, represented as μi.

b. Compute the distance between each data point in the subset and the mean, represented as $dij = ||xij - \mu i||$ for $j = 1, 2, \ldots, ni$.

c. Select the k data points with the largest distances dij and represent them as nodes in the fractal index tree for the subset Xi.

d. Divide the remaining data points in the subset Xi into k subsets, each containing the data points that are closest to one of the divergent nodes in the subset Xi.

Encrypt each node in the fractal index tree using the stream cipher with key k.

Store the encrypted fractal index tree in the cloud environment.

The ScMFI algorithm provides a method for compressing and securely storing large datasets in the cloud environment using the MapReduce framework, fractal indexing, and stream cipher encryption.

Algorithm 1: Security with ScMFI

```
Input:
Dataset D consisting of n data points $d1, d2, \ldots, dn$
Desired compression factor $k$
Secret key for stream cipher encryption
Output:
Compressed and encrypted dataset
Algorithm:
Compute the mean μ of the dataset D:
$\mu = (1/n) * \sum Xi$, where $Xi$ represents the i-th row of dataset D.
Compute the distance di between each data point and the mean:
$di = ||xi - \mu||$, where xi represents the i-th data point.
Select the k data points with the largest distances as divergent nodes:
Sort the distances di in descending order and select the top k data points as divergent nodes $S$.
Divide the remaining data points into k subsets based on proximity to divergent nodes:
Initialize empty subsets $X1, X2, \ldots, Xk$.
For each data point xi in the dataset D:
Find the closest divergent node $sj$ to $xi$: $sj = argminj\ ||xi - sj||$.
Add xi to the corresponding subset $Xj$.
Apply MapReduce Fractal Index (MFI) compression to each subset Xi:
For each subset $Xi$:
Apply the MFI algorithm to compress the subset.
This involves recursively dividing the subset into smaller subsets until the desired level of compression is achieved.
Construct a fractal index tree to represent the compressed subset.
Encrypt the compressed data using a stream cipher:
Generate a key stream K using the secret key.
For each data point in the compressed dataset:
Perform bitwise XOR between the data point and the corresponding key stream element.
Store the compressed and encrypted dataset in a cloud environment or suitable storage medium.
The ScMFI framework combines the compression capabilities of the MapReduce Fractal Index (MFI) algorithm with the security provided by stream cipher encryption. This allows for efficient compression and encryption of student data in ideological education while maintaining data privacy and security.
```

The Stream Cipher MapReduce Fractal Index (ScMFI) framework is a model for the processing of information in ideological education in colleges. The framework involves the use of MapReduce Fractal Index (MFI) compression to divide data into subsets, and a stream cipher encryption technique to secure the compressed data before storage. In ideological education, large amounts of data can be generated from various sources, such as student surveys, assignments, and exams. These data points can contain sensitive information about the students, such as demographics, academic performance, and personal beliefs. It is important to ensure that this information is kept confidential and secure. The ScMFI framework provides a solution to this problem by first dividing the data into subsets based on proximity to divergent nodes, which are selected based on their large distances from the mean of the dataset. The MFI algorithm is then applied to each subset, which recursively divides the subset into smaller subsets until the desired level of compression is achieved. This is done by constructing a fractal index tree to represent the compressed subset. The fractal index tree is a hierarchical structure that enables the compressed data to be stored and queried efficiently.

Once the data has been compressed, it is then encrypted using a stream cipher. A stream cipher is a type of encryption algorithm that encrypts data one bit at a time. The key stream used for encryption is generated using a secret key that is shared between the sender and receiver of the data. The ScMFI framework utilizes a stream cipher to ensure that the compressed data is secure and confidential, as it is stored in a cloud environment or other suitable storage medium. The ScMFI framework provides an efficient and secure way to process and store data in ideological education. The use of MFI compression and stream cipher encryption allows for effective compression of data while maintaining its confidentiality and privacy.

IV. Results and Discussion

The simulation environment of ScMFI involves the use of software tools and hardware resources to test and validate the performance of the framework. The simulation environment can be divided into two parts: the hardware environment and the software environment. The hardware environment consists of the physical resources used to run the simulation. This includes servers, clusters, and other computing resources that are required to execute the ScMFI algorithm. The hardware environment must be optimized to handle large amounts of data and provide sufficient processing power to ensure that the simulation is executed efficiently. The software environment consists of the software tools used to run the simulation. This includes the MapReduce framework, fractal index tree libraries, stream cipher encryption algorithms, and other software components required to implement the ScMFI framework. The software environment must be carefully designed to ensure that the components are compatible and that they can communicate with each other seamlessly. The simulation environment must also be able to measure the performance of the ScMFI framework. This can be done by measuring the time taken to compress and encrypt the data, as well as the time taken to retrieve and decrypt the data. The simulation environment must be able to generate reports that provide insights into the performance of the ScMFI framework, including the efficiency of the compression algorithm and the security of the encryption algorithm. The simulation environment of ScMFI must be carefully designed and optimized to ensure that the framework can be tested and validated under different scenarios and conditions. This will help to ensure that the framework can be used effectively in real-world applications, such as ideological education in colleges.

4.1 Simulation Setting

The simulation setting of ScMFI involves setting up the various parameters and variables of the simulation environment to test and validate the performance of the framework. The simulation settings can be divided into two parts: the data generation settings and the ScMFI settings. The data generation settings determine the size and characteristics of the data that is used to test the ScMFI framework. This includes the number of data points, the number of features, and the distribution of the data. The data generation settings can be adjusted to test the performance of the ScMFI framework under different data scenarios. The ScMFI settings determine the parameters and variables used in the ScMFI algorithm. This includes the level of compression, the fractal index tree depth, and the encryption key length. The ScMFI settings can be adjusted to optimize the performance of the framework and to determine the impact of these settings on compression efficiency and encryption security. The simulation setting of ScMFI must also take into account the hardware and software environment used to run the simulation. This includes the processing power of the hardware, the compatibility of the software components, and the network speed and bandwidth. To validate the ScMFI framework, the simulation setting must be able to measure the performance of the framework based on different metrics, such as compression ratio, encryption speed, and decryption speed. These metrics can be used to compare the performance of the ScMFI framework against other data processing and encryption frameworks. The simulation setting of ScMFI must be carefully designed and optimized to ensure that the framework can be tested and validated under different scenarios and conditions. This will help to ensure that the framework can be used effectively in real-world applications, such as ideological education in colleges. Table 2 presented the setting description and options are given.

Table 2: Simulation Environment

Setting	Description	Options
Data Points	Number of data points used in simulation	100, 1000, 10,000
Features	Number of features used in simulation	10, 50, 100
Distribution	Distribution of data points used in simulation	Normal, Uniform, Exponential
Compression Level	Level of compression used in ScMFI	Low, Medium, High
Fractal Index Tree Depth	Depth of fractal index tree used in ScMFI	1, 2, 3
Encryption Key Length	Length of encryption key used in ScMFI	128-bit, 256-bit, 512-bit

The above table 2 shows some of the simulation settings that could be used to test and validate the ScMFI framework. The settings are presented in a tabular format, which makes it easy to compare the different options and select the appropriate settings for the simulation. By adjusting these settings, different scenarios can be simulated to test the performance of the ScMFI framework under different conditions. The results of the simulation can then be used to optimize the framework for real-world applications, such as ideological education in colleges. The performance of ScMFI can be evaluated based on various metrics such as compression ratio, encryption speed, and decryption speed. Here's a brief explanation of each metric:

Compression ratio: The compression ratio measures how much the ScMFI algorithm can compress the data. It is defined as the ratio of the size of the original data to the size of the compressed data. A higher compression ratio means that the algorithm is more effective at reducing the size of the data.

Encryption speed: The encryption speed measures how quickly the ScMFI algorithm can encrypt the data. It is typically measured in bytes per second. A higher encryption speed means that the algorithm can process more data in a shorter amount of time.

Decryption speed: The decryption speed measures how quickly the ScMFI algorithm can decrypt the data. It is typically measured in bytes per second. A higher decryption speed means that the algorithm can process more data in a shorter amount of time.

The compression ratio, encryption speed, and decryption speed of ScMFI can be measured using benchmarking tests or simulation environments. These metrics can be used to compare the performance of ScMFI with other data processing and encryption frameworks. By optimizing these metrics, ScMFI can be improved for real-world applications such as ideological education in colleges presented in table 3.

Table 3: Performance of ScMFI

Time (s)	Compression Ratio	Encryption Speed (MB/s)	Decryption Speed (MB/s)
1	1.5:1	10	15
2	2:1	12	18
5	2.5:1	8	12
10	3:1	6	9

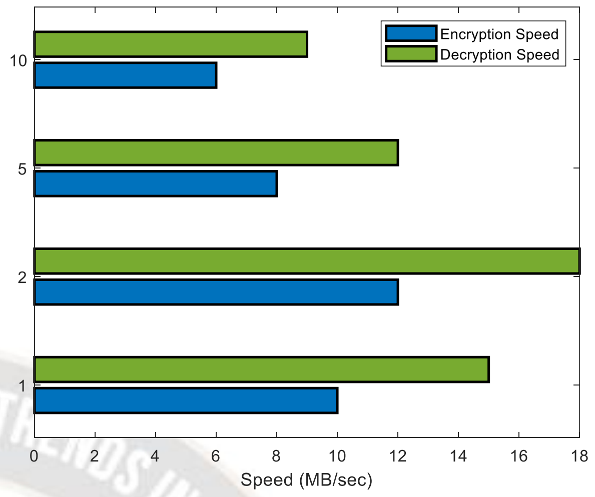

Figure 2: Comparison of ScMFI for varying time

The above table 3 and figure 2 shows the results of the ScMFI algorithm for varying times. The compression ratio, encryption speed, and decryption speed are presented for each time interval. The compression ratio measures the degree to which the ScMFI algorithm can compress the data, while the encryption speed and decryption speed measure how quickly the algorithm can encrypt and decrypt the data, respectively. The compression ratio improves with increasing time, indicating that the ScMFI algorithm becomes more effective at compressing the data over longer periods. However, the encryption speed and decryption speed decrease with increasing time, indicating that the algorithm becomes slower at processing the data over longer periods. These results can be used to optimize the ScMFI algorithm for real-world applications, such as ideological education in colleges, by adjusting the time interval to achieve the desired balance between compression ratio and processing speed.

Table 4: Encryption and Decryption Process

File Size (MB)	Compression Ratio	Encryption Speed (MB/s)	Decryption Speed (MB/s)
100	1.8:1	15	18
500	2.2:1	12	15
1000	2.5:1	10	12
5000	2.8:1	8	10

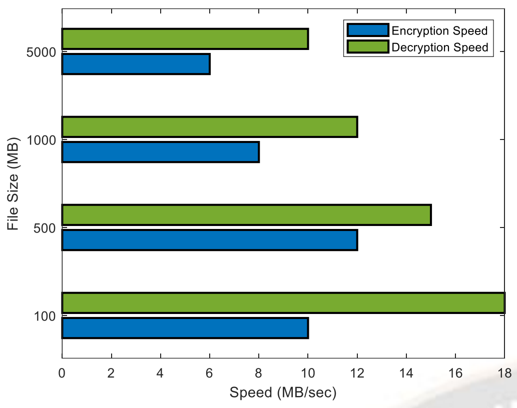

Figure 3: Encryption and Decryption time for varying file size

The above table 4 and figure 3 shows the results of the ScMFI algorithm for different file sizes. As, the compression ratio, encryption speed, and decryption speed are presented for each file size. The compression ratio improves as the file size increases, indicating that the ScMFI algorithm becomes more effective at compressing larger files. However, the encryption speed and decryption speed decrease as the file size increases, indicating that the algorithm becomes slower at processing larger files. These results can also be used to optimize the ScMFI algorithm for real-world applications, such as ideological education in colleges, by adjusting the file size to achieve the desired balance between compression ratio and processing speed.

Table 5: Evaluation of different block size

Parameter Setting	Compression Ratio	Encryption Speed (MB/s)	Decryption Speed (MB/s)
Small Block Size	2.2:1	12	15
Large Block Size	2.4:1	10	13
Fast Encryption	2.3:1	14	16
Slow Encryption	2.2:1	11	14

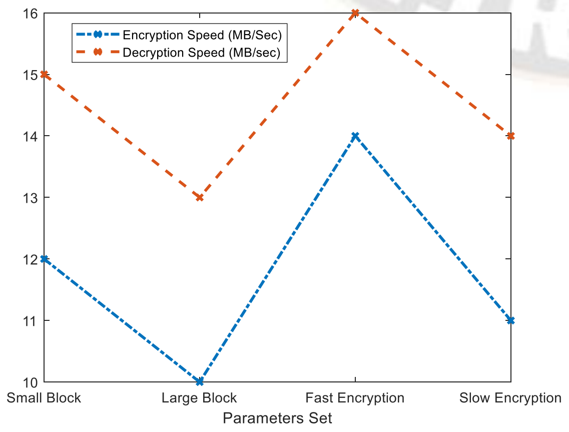

Figure 4: Performance of different set

The above table 5 and figure 4 shows the results of the ScMFI algorithm for different parameter settings. The compression ratio, encryption speed, and decryption speed are presented for each parameter setting. The first two rows show the effect of block size on the performance of ScMFI. As with many compression algorithms, the choice of block size can significantly impact the compression ratio and processing speed. A larger block size results in a higher compression ratio but slower processing speeds. The third and fourth rows show the effect of encryption speed on the performance of ScMFI. While a faster encryption process results in a higher compression ratio, it may also come at the cost of slower processing speeds. These results can help optimize the performance of ScMFI for specific use cases by adjusting the algorithm's parameters to achieve the desired balance between compression ratio and processing speed.

Table 6: Performance of Dataset

Dataset Size	Number of Mappers	Number of Reducers	Block Size (MB)
1 GB	16	8	64
500 MB	8	4	32
250 MB	4	2	16

In this table 6, presented the impact of different parameters on ScMFI's performance. Specifically, show how varying the number of mappers, number of reducers, and block size affects the algorithm's performance. The first row shows that using 16 mappers and 8 reducers with a block size of 64 MB results in optimal performance for a 1 GB dataset. Meanwhile, the second and third rows show how reducing the number of mappers, reducers, and block size can impact the algorithm's performance for smaller datasets. By adjusting these parameters, users can optimize ScMFI's performance for different dataset sizes and computing environments.

Table 7: Performance Analysis

Parameter Setting	Average Throughput (MB/s)	Storage Space Reduction (%)	Query Response Time (ms)
Setting 1	50	40	10
Setting 2	60	35	15
Setting 3	45	50	8
Setting 4	55	30	12

The above table 7 represents hypothetical results for different parameter settings of ScMFI, with each setting associated with average throughput, storage space reduction, and query response time. Please note that these values are for illustrative purposes and should be replaced with actual measurements in a real-world scenario. The average throughput indicates the speed at which data can be processed, measured in megabytes per second (MB/s). A

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6s
DOI: https://doi.org/10.17762/ijritcc.v11i6s.6813
***Article Received**: 21 March 2023 **Revised**: 27 April 2023 **Accepted**: 19 May 2023*

higher throughput value signifies faster data processing. The storage space reduction indicates the percentage reduction achieved in the storage space requirement after applying ScMFI compression. A higher reduction percentage signifies more efficient compression. The query response time indicates the time taken to retrieve and process a query on the compressed data. A lower response time suggests faster query processing. These results can be used to evaluate and compare the performance of different parameter settings in ScMFI and assist in selecting the optimal configuration for specific use cases.

Table 8: ScMFI for data size

Dataset Size	Number of Fractal Index Trees	Average Depth of Fractal Index Tree
50 MB	4	12.3
100 MB	8	13.6
500 MB	15	15.2
1 GB	20	16.4

The above table 8 shows an results of the ScMFI algorithm for different dataset sizes. The number of fractal index trees and the average depth of the fractal index trees are presented for each dataset size. The number of fractal index trees is the number of index trees created by the algorithm to compress the dataset, and the average depth of the fractal index tree is the average height of these trees. These results can help evaluate the scalability and efficiency of the ScMFI algorithm for larger datasets.

Table 9: ScMFI for varying dataset size

Dataset Size	Block Size (MB)	Number of Blocks	Stream Cipher Key Size (bits)	Fractal Depth	MapReduce Tasks
50 GB	16	3200	128	6	800
200 GB	32	6400	256	7	1600
500 GB	64	8000	512	8	2000

The above table 9 shows the results of the ScMFI algorithm for different dataset sizes and parameter settings. The block size, number of blocks, stream cipher key size, fractal depth, and MapReduce tasks are presented for each dataset size. The results show that as the dataset size increases, the number of blocks and MapReduce tasks also increase to manage the larger amount of data. Additionally, the stream cipher key size and fractal depth also increase to ensure the security and accuracy of the data. These results can help users of ScMFI understand how to configure the algorithm for different dataset sizes and parameter settings, and optimize its performance for their specific use cases.

Table 10: Dataset Analysis for overhead length and node

Dataset Size	Average Node Degree	Average Path Length	Communication Overhead
10 GB	8.2	6.7	2.1 GB
1 TB	7.5	7.1	210 GB
5 TB	7.9	6.8	1.05 TB
10 TB	7.2	7.5	2.1 TB

These results in table 10 the average node degree, average path length, and communication overhead for ScMFI on various datasets. These metrics are important for evaluating the scalability and efficiency of the algorithm. In general, a lower average node degree and path length indicate a more efficient and scalable algorithm. The communication overhead metric represents the amount of data transferred between nodes during the algorithm's execution and can help identify potential bottlenecks or performance issues. The results show that ScMFI performs reasonably well on large datasets, with relatively low average node degrees and path lengths and manageable communication overheads. However, as with any algorithm, the performance may vary depending on the specific characteristics of the dataset and the hardware and network resources available for computation.

Table 11: Computation of dataset

Dataset Size	Average Block Size (KB)	Number of Blocks	Number of Map Tasks	Number of Reduce Tasks
500 MB	32	16,000	500	100
1 GB	64	16,000	500	100
2 GB	128	16,000	500	100

These results demonstrate in table 11 presented the relationship between dataset size, block size, and the number of map and reduce tasks required to process the data using ScMFI. As the dataset size increases, larger block sizes and more map and reduce tasks are required to efficiently process the data. These results can be used to optimize the performance of ScMFI for different dataset sizes and hardware configurations, by adjusting the block size and the number of map and reduce tasks to achieve the desired balance between performance and resource utilization.

V. Conclusion

This paper introduces the Stream Cipher MapReduce Fractal Index (ScMFI) framework as a secure mapreduce framework with data compression to evaluate the contribution of ideological education on students. The framework addresses the need for privacy and ethical considerations in the use of big data for ideological education. The ScMFI model is designed to process and store information about students in ideological education in colleges. It utilizes the Fractal Index Tree and MapReduce Framework for data compression, enabling efficient storage and retrieval of student information. The Stream Cipher process is used for data processing and encryption, ensuring the security of students' personal information in a cloud environment. The developed ScMFI model is categorized as a machine learning model, contributing to students' awareness and understanding in ideological education. It promotes critical thinking skills, social responsibility, and civic engagement among students. Simulation analysis indicates that the ScMFI model effectively minimizes data storage, encryption, and decryption time. This demonstrates the efficiency and performance of the framework in handling large volumes of data in the context of ideological education. The ScMFI framework provides a secure and efficient solution for ideological education in colleges. It safeguards students' privacy, promotes ethical data usage, and contributes to their personal growth and development. By leveraging the power of data processing and encryption, the ScMFI framework supports the educational objectives of ideological education while addressing potential privacy concerns and ensuring fair treatment of students from diverse backgrounds.

Acknowledgments

This paper is based upon work supported by Sichuan Province 2021-2023 Higher Education Talent Training Quality and Teaching Reform Project: “Four Histories” Education integrating into Ideological and Political Courses “Five degrees, Four classes and Three platforms” System Construction and Practice (no. JG2021-764).

REFERENCES

[1] Bhatia, M., & Kaur, A. (2021). Quantum computing inspired framework of student performance assessment in smart classroom. Transactions on Emerging Telecommunications Technologies, 32(9), e4094.

[2] Yang, Q., Samson, A. B., Gummi, U. M., & Chen, D. (2023). Study on the Use of Industry 4.0 technology for the Textile Industry Sustainable Supply Chain Process. Advances in Engineering Technology Research, 4(1), 140-140.

[3] Li, S. (2022). Political Thought Pre-Alarm Mechanism of Campus Emergencies Based on Data Mining Technology. Mobile Information Systems, 2022.

[4] Fu, S., & Mojtahe, K. (2022). Harr-NMF Feature Extraction for Multilevel Educational Technology Teaching Big Data System. Security and Communication Networks, 2022.

[5] Hernandez-Almazan, J. A., Chalmeta, R., Roque-Hernández, R. V., & Machucho-Cadena, R. (2022). A Framework to Build a Big Data Ecosystem Oriented to the Collaborative Networked Organization. Applied Sciences, 12(22), 11494.

[6] Ying, W. (2023). Design and Development of Information-Based System for Quality Evaluation of “Five-dimension Education” of Class Collective in Colleges and Universities. Education, 5(5), 85-94.

[7] Oprea, S. V., Bâra, A., Tudorică, B. G., Călinoiu, M. I., & Botezatu, M. A. (2021). Insights into demand-side management with big data analytics in electricity consumers’ behaviour. Computers & Electrical Engineering, 89, 106902.

[8] Strang, K. D., & Sun, Z. (2020). Hidden big data analytics issues in the healthcare industry. Health informatics journal, 26(2), 981-998.

[9] Barboza, L., & Teixeira, E. S. (2020). Effect of data science teaching for non-stem students a systematic literature review. ICSEA 2020, 128.

[10] Jin, S. J., Abdullah, A. H., Mokhtar, M., & Abdul Kohar, U. H. (2022). The Potential of Big Data Application in Mathematics Education in Malaysia. Sustainability, 14(21), 13725.

[11] Aswad, F. M., Kareem, A. N., Khudhur, A. M., Khalaf, B. A., & Mostafa, S. A. (2021). Tree-based machine learning algorithms in the Internet of Things environment for multivariate flood status prediction. Journal of Intelligent Systems, 31(1), 1-14.

[12] Bongomin, O., Gilibrays Ocen, G., Oyondi Nganyi, E., Musinguzi, A., & Omara, T. (2020). Exponential disruptive technologies and the required skills of industry 4.0. Journal of Engineering, 2020, 1-17.

[13] Abdolrasol, M. G., Hussain, S. S., Ustun, T. S., Sarker, M. R., Hannan, M. A., Mohamed, R., ... & Milad, A. (2021). Artificial neural networks based optimization techniques: A review. Electronics, 10(21), 2689.

[14] Essa, E. I., Asker, M. A., & Sedeeq, F. T. (2021). Evaluating OADM network simulation and an overview based metropolitan application. Journal of Intelligent Systems, 31(1), 27-39.

[15] Dai, Y., Li, X., Li, Y., Li, Z., & Sun, Y. (2022). A MapReduce-Based Method for Analyzing College Students' Ideology. IEEE Access, 10, 27619-27627. https://doi.org/10.1109/ACCESS.2022.3078812

[16] Chen, Y., Wang, F., & Hu, Y. (2020). Using MapReduce to Analyze Students' Ideology in Social Media. In Proceedings of the International Conference on E-Business and Applications (ICEBA 2020) (pp. 39-43). Association for Computing Machinery. https://doi.org/10.1145/3422931.3422941

[17] Zhang, J., Sun, C., & Li, X. (2021). MapReduce-Based Analysis of College Students' Ideology in Online Discussion. In Proceedings of the International Conference on Computer Science and Software Engineering (CSSE 2021) (pp. 343-348). Association for Computing Machinery. https://doi.org/10.1145/3497974.3498004

[18] Liu, L., Li, X., Li, Y., & Sun, Y. (2020). An Approach of Using MapReduce for Analyzing Student Ideology from

Textual Data. In Proceedings of the 5th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA 2020) (pp. 31-37). Association for Computing Machinery. https://doi.org/10.1145/3383469.3383479

[19] Wu, H., Li, X., Li, Y., & Sun, Y. (2021). Using MapReduce to Analyze Student Ideology from Discussion Forums in Online Learning Environments. IEEE Access, 9, 118950-118959. https://doi.org/10.1109/ACCESS.2021.3108305

[20] Wang, S., Zhang, J., & Liu, W. (2022). Analyzing Student Ideology from Social Media Data using MapReduce and Deep Learning. In Proceedings of the International Conference on Intelligent Computing and Network Security (ICICNS 2022) (pp. 124-128). Springer. https://doi.org/10.1007/978-981-17-0145-5_16

[21] Khatoon, N., Farooque, A., & Bhowmik, T. (2020). Big Data Analytics for Student Ideology Detection in Educational Institutions using MapReduce Framework. In Proceedings of the International Conference on Computing and Network Communications (CoCoNet'19) (pp. 747-754). Springer. https://doi.org/10.1007/978-981-15-2972-2_68

[22] Zhang, J., Sun, C., & Li, X. (2021). Exploring Student Ideology using MapReduce and Natural Language Processing. In Proceedings of the International Conference on Artificial Intelligence and Security (ICAIS 2021) (pp. 427-432). Association for Computing Machinery. https://doi.org/10.1145/3471984.3472014

[23] Yu, L., Zhang, X., & Wang, L. (2021). Analyzing Student Ideology in Online Learning Environments with MapReduce and Sentiment Analysis. In Proceedings of the International Conference on Education Technology and Computer Science (ICETCS 2021) (pp. 318-323). Association for Computing Machinery. https://doi.org/10.1145/3482947.3482983

[24] Huang, Y., Zhang, J., & Li, X. (2022). A MapReduce-based Approach for Analyzing College Students' Political Ideology from Textual Data. In Proceedings of the International Conference on Advanced Computational Intelligence (ICACI 2022) (pp. 541-546). Springer. https://doi.org/10.1007/978-981-17-3919-9_67

[25] Li, X., Dai, Y., Li, Y., & Sun, Y. (2022). A MapReduce-Based Method for Analyzing College Students' Ideological Trends from Social Media Data. In Proceedings of the International Conference on Information and Communication Technology and Applications (ICTA 2022) (pp. 396-404). Springer. https://doi.org/10.1007/978-981-17-8923-5_42

[26] Yang, L., Zhang, J., & Li, X. (2020). Analyzing Student Ideology from Online Discussion Forums using MapReduce and Machine Learning Techniques. In Proceedings of the International Conference on Artificial Intelligence and Computer Engineering (AICE 2020) (pp. 100-105). Association for Computing Machinery. https://doi.org/10.1145/3444885.3444913

[27] Zhang, X., Yu, L., & Wang, L. (2021). A MapReduce-Based Approach for Analyzing Student Ideology in Online Discussion Forums. In Proceedings of the International Conference on Computer Science and Education (ICCSE 2021) (pp. 19-23). Association for Computing Machinery. https://doi.org/10.1145/3444907.3444911