Generative Architecture for Data Imputation in Secure Blockchain-enabled Spatiotemporal Data Management

Song Li[1], WenFen Liu[1,*], Yan Wu[2] and Jie Zhao[1]

[1]*School of Computer and Information Security, Guilin University of Electronic Technology, Guilin, China*
[2]*Unit 95795 of PLA, Guilin, China*
E-mail: 21031102007@mails.guet.edu.cn; liuwenfen@guet.edu.cn; jewel.wu@163.com; 21032303181@mails.guet.edu.cn
[*]*Corresponding Author*

Received 31 August 2023; Accepted 17 October 2023;
Publication 27 March 2024

Abstract

In the era of big data, one of the most critical challenges is ensuring secure access, retrieval, and sharing of linked spatiotemporal data. To address this challenge, this paper introduces a groundbreaking blockchain-enabled evolutionary indirect feedback graph algorithm for the secure management of interconnected spatiotemporal datasets. The algorithm utilizes a generative neural network model for data imputation, predicting and generating plausible values to improve dataset completeness and integrity. The core architecture utilizes blockchain technology to optimize data retrieval efficiency and uphold robust access control mechanisms. The algorithm incorporates indirect feedback mechanisms, allowing users to provide implicit feedback through their interactions, enhancing the relevance and efficiency of data retrieval. In addition. sophisticated graph-based techniques are used to model intricate relationships between data entities, facilitating seamless data retrieval

Journal of Web Engineering, Vol. 23_1, 111–164.
doi: 10.13052/jwe1540-9589.2315

and sharing in interwoven datasets. The algorithm's data security approach includes comprehensive access control mechanisms, encryption, and authentication mechanisms, safeguarding data confidentiality and integrity. Extensive evaluations show significant enhancements in retrieval performance and access control precision, making the proposed model a promising solution for the secure management of expansive interconnected spatiotemporal data.

Keywords: Indirect feedback graph algorithm, linked spatiotemporal data, big data analysis, secure access, data retrieval, generative AI, blockchain.

1 Introduction

In the era of big data, the volume and complexity of information have grown exponentially, presenting new challenges in managing and extracting value from massive datasets [1]. One significant aspect of this challenge is ensuring secure access, retrieval, and sharing of linked spatiotemporal data. The interconnected nature of these datasets, combined with the sensitive nature of their information, calls for robust solutions that address both efficiency and security concerns [2].

The proliferation of linked spatiotemporal data sources, such as geographical and temporal datasets, has revolutionized various domains, including transportation, urban planning, environmental monitoring, and healthcare. These datasets provide valuable insights by capturing dynamic relationships and patterns over time and space. However, their enormous size and interconnectedness pose unique challenges in managing, retrieving, and securely sharing data [3].

Data security is of utmost importance in today's interconnected world. Organizations and individuals must ensure that sensitive information remains protected from unauthorized access, data breaches, and malicious activities. Furthermore, with linked spatiotemporal data, the challenge extends beyond traditional security concerns, as maintaining the integrity and privacy of the data while enabling efficient retrieval and sharing becomes paramount [4].

Existing solutions for secure data management often fall short of effectively addressing the intricacies of big-linked spatiotemporal data. Traditional access control mechanisms and retrieval algorithms struggle to handle these datasets' scale, complexity, and dynamic nature. Moreover, incorporating user feedback and preferences to enhance retrieval relevance and adaptability remains an ongoing challenge [5].

Incomplete or missing data is a critical challenge that arises when dealing with real-world data, particularly in the context of spatiotemporal datasets. In many cases, due to various reasons such as sensor errors, data transmission issues, or even limitations in data collection processes, certain data points might lack essential information. This can lead to gaps in the dataset that could potentially impact the quality, accuracy, and reliability of subsequent analyses and applications.

Addressing this challenge is crucial for making informed decisions and drawing meaningful insights from the data. Approaches such as data imputation, which involves predicting and filling in missing values, play a significant role in enhancing the integrity and completeness of such datasets. By acknowledging this challenge, researchers and practitioners have focused on developing strategies and techniques that effectively handle missing data to ensure the validity of their findings and conclusions [6–8]. This paper responds to this challenge by introducing a pioneering approach that not only harnesses the potential of blockchain technology but also integrates advanced generative neural network models to address the prevalent issue of incomplete or missing data in such datasets [9].

This research study introduces a technique for secure management of interconnected spatiotemporal datasets, utilizing a blockchain-enabled evolutionary indirect feedback graph. The technique incorporates a generative neural network model to enhance dataset completeness and integrity by predicting and generating plausible values. With the integration of blockchain technology, the design aims to improve data retrieval performance and establish robust access control measures. The system incorporates indirect feedback techniques, enabling users to contribute implicit input through their interactions, thereby enhancing data retrieval efficiency and relevance. Additionally, advanced graph-based approaches describe detailed interactions between data items, facilitating seamless data retrieval and sharing across interconnected datasets. To ensure data security, the algorithm incorporates extensive access control features, encryption, and authentication systems, safeguarding data confidentiality and integrity.

2 Methodology

At its essence, the proposed method introduces a groundbreaking blockchain-enabled evolutionary indirect feedback graph algorithm [10], meticulously tailored for the secure management of vast interconnected spatiotemporal

datasets. This algorithm resides at the nexus of several innovations, each aimed at enhancing data quality, security, and usability [11].

Central to this approach is the strategic use of a generative neural network [12, 13] model designed explicitly for data imputation. In scenarios where data points are marred by missing or incomplete attributes, this model steps in to predict and generate plausible values, thereby enhancing the completeness and integrity of the dataset. By filling in these gaps, the generative neural network contributes to creating a more comprehensive and reliable foundation for subsequent analyses [14].

Complementing this data-imputation facet is the core architecture's utilization of blockchain technology [15, 16]. This empowers the algorithm with the capability to optimize data retrieval efficiency while concurrently upholding robust access control mechanisms. The integration of an evolutionary nature further equips the algorithm to dynamically adapt to the fluidity of changing data environments, thus ensuring sustained peak performance and responsiveness.

Furthermore, a pivotal facet of this algorithm lies in its incorporation of indirect feedback mechanisms, allowing users to provide implicit feedback through their interactions [10]. This invaluable feedback loop acts as a guiding compass, continuously refining and personalizing the data retrieval process. As a result, the relevance and efficiency of data retrieval are profoundly elevated, aligning the algorithm's outcomes more closely with users' preferences and intentions.

To enhance the capabilities of the algorithm, advanced graph-based techniques are employed to represent complex relationships among data entities. As a result, this enables effortless retrieval and sharing of data, particularly within the intricate interconnections of datasets. Of utmost importance, the algorithm demonstrates its strength through a robust approach to data security. This involves implementing comprehensive access control mechanisms, which empower data owners to intricately define and enforce access policies based on user roles and permissions. Such fortification guarantees the resilience of data against breaches, ensuring that only authorized individuals can access it.

Moreover, the algorithm bolsters its security stance by incorporating encryption and authentication mechanisms [17]. These protective layers safeguard the confidentiality and integrity of data during retrieval and sharing endeavors, erecting additional barriers against unauthorized access and tampering.

The effectiveness of this innovative approach is substantiated through empirical assessments, shedding light on its notable efficiency and effectiveness when compared to advanced benchmarks. These outcomes highlight significant improvements in retrieval performance and precision for access control, consolidating its status as an exceptionally promising solution for securely managing extensive interconnected spatiotemporal data.

With its dynamic evolutionary nature and a mechanism for integrating indirect feedback, the algorithm is well-equipped not only to navigate but also to thrive within the dynamic contours of evolving data landscapes. When confronted with the challenges arising from the relentless growth of linked spatiotemporal data, the algorithm emerges as a formidable contender. By synergizing with blockchain technology's capabilities, it establishes an innovative and resilient paradigm for secure data management in the constantly expanding realm of big data. Appendix 1 presents pseudocode 1, which represents the algorithm based on the provided description.

This pseudocode delineates the fundamental steps and functionalities of the proposed blockchain-enabled evolutionary indirect feedback graph algorithm. It initiates with population initialization and fitness evaluation. The evaluateFitness() function calculates the fitness level of each individual within the population, expressing how proficiently they solve the given problem. Moreover, this pseudocode introduces the process of generative neural network-based data imputation. By invoking the imputeMissingData() function, it performs the task of imputing missing data for each individual's solution. The imputed data is then stored in the individual.imputed_data attribute. The function imputeMissingData() manages the data imputation process by utilizing a generative neural network. It first extracts the incomplete data from the current individual's solution and then trains a generative neural network model (trainGenerativeModel(data)) using the available complete data. Once trained, the generative model generates imputed values for the missing attributes in the incomplete data (generateImputedValues(incomplete_data)). These imputed values are subsequently merged with the incomplete data, resulting in a complete imputed dataset (mergeImputedData(imputed_values, incomplete_data)), which is returned.

After the evolutionary process, the best individual is selected. The algorithm constructs a graph representation of the data and initializes the access control mechanisms. Then, it enters another loop to handle user interactions. It obtains the user query, infers indirect feedback based on user behavior and contextual information, and fine-tunes the query using this feedback.

The algorithm performs graph traversal on the constructed graph to identify relevant data entities and applies access control filtering based on user permissions and contextual factors. The filtered data is then displayed to the user. Finally, the algorithm encrypts and authenticates the data to ensure secure retrieval and sharing operations [8, 15, 16].

Overall, the proposed blockchain-enabled evolutionary indirect feedback graph algorithm provides a comprehensive solution for secure big linked spatiotemporal data retrieval, sharing, and access control. By combining evolutionary computation, indirect feedback, and graph-based techniques, it offers an efficient and adaptable approach to managing and extracting value from interconnected datasets while maintaining stringent security measures.

2.1 Description of the Algorithm's Components and Mechanisms

The proposed blockchain-enabled evolutionary indirect feedback graph algorithm consists of several key components and mechanisms that work together to optimize the secure retrieval, sharing, and access control of big linked spatiotemporal data. These components and mechanisms are designed to enhance efficiency, relevance, and security throughout the data management process.

It must be noted that by combining the following components and mechanisms, the algorithm optimizes the retrieval process, enhances relevance through indirect feedback, enables efficient navigation of linked spatiotemporal data through graph-based modeling, and ensures secure access control through fine-grained permissions, encryption, and authentication. This comprehensive approach contributes to the efficient and secure management of big linked spatiotemporal data.

2.1.1 Genetic algorithm-based optimization

The algorithm utilizes a genetic algorithm or other evolutionary optimization techniques to iteratively improve the data retrieval process. By employing evolutionary principles such as selection, crossover, and mutation, the algorithm evolves a population of search strategies over multiple generations. This optimization process aims to find the most effective and efficient retrieval strategies that adapt to the evolving data environment and user preferences [17, 18].

The pseudocode 2 presents the genetic algorithm-based optimization method. The pseudocode outlines the basic structure of a genetic algorithm optimization process: initialization, main loop, selection of parents, offspring

generation, evaluation, and replacement. It checks if the termination condition is met, exits the loop, or repeats the main steps. The best individual is then obtained from the final population based on its fitness value, representing the optimal or near-optimal solution to the problem.

During the execution of the genetic algorithm, the population evolves over generations, with individuals being selected, recombined, and mutated to explore the solution space. Through the iterative process, the algorithm tends to converge towards a population of individuals with higher fitness, which represents a near-optimal or optimal solution to the problem [19, 20].

2.1.2 Indirect feedback mechanism

The algorithm incorporates indirect feedback mechanisms, allowing users to provide implicit feedback through their interactions with the system. Instead of explicit feedback, the algorithm infers user preferences based on user behavior, query patterns, and contextual information. This indirect feedback is used to continuously refine the retrieval process, adapting it to the user's evolving needs and improving the relevance of the retrieved data [5, 21].

The following is a description of an algorithm denoted as pseudocode 3, which incorporates an indirect feedback mechanism. This algorithm utilizes a series of steps involving data retrieval, user interaction, query refinement, data display, user feedback collection, retrieval process adjustment, feedback analysis, and retrieval process finalization. It also encompasses the collection, analysis, and adjustment of user feedback to enhance the relevance of retrieved data and cater to user requirements. The algorithm prioritizes user-friendliness and efficiency to ensure an optimal user experience.

Moreover, the algorithm continually enhances the retrieval process by deducing user preferences from their behavior, query patterns, and contextual information. It then adapts the retrieval process accordingly to augment the relevancy of the obtained data. In addition, the feedback obtained from user interactions further refines and optimizes the retrieval process, guaranteeing its alignment with the evolving needs of the user.

2.1.3 Graph-based modelling

The algorithm employs graph-based modeling techniques to represent the interconnected relationships between data entities in the linked spatiotemporal data. By constructing a graph representation, the algorithm captures the dependencies, similarities, and hierarchical structures present in the data. Graph traversal algorithms, similarity measures, and clustering techniques are used to navigate the graph, identify relevant data entities, and optimize

the retrieval process. This graph-based approach enhances the efficiency and effectiveness of data retrieval and sharing [7, 12].

The pseudocode 4 represents the algorithm that employs graph-based modeling techniques. The algorithm uses graph-based modeling techniques to construct a graph representation of linked spatiotemporal data, capture dependencies, similarities, and hierarchical structures between data entities. It then uses a user interaction loop to continuously interact with the user and obtain a query. The graph traversal is used to identify relevant data entities, and similarity measures are calculated to assess their relevance. Clustering techniques are performed on the data entities, grouping them based on characteristics or attributes. The results are displayed to the user, and clusters of relevant data entities are presented.

The algorithm employs graph traversal algorithms to navigate the graph and identify relevant data entities. Similarity measures are calculated to assess the similarity between different entities, aiding in determining their relevance. Clustering techniques are then applied to group similar entities together, enhancing the efficiency and effectiveness of data retrieval and sharing. To further elaborate on this point, we can assume that the likeness between a user and their nearby connections within the reliable network is determined by an unidentified distribution function that gauges the proximity of two neighbors' evaluative actions [22–24]. Define:

$$\lambda(G) = \max_{x \perp u} \frac{\| Mx \|}{\| x \|} = \max_{x \perp u} \frac{|\langle Mx, x \rangle|}{\langle x, x \rangle}.$$

$\lambda(G)$: This represents the eigenvalue of a graph G.
$x \perp u$: This indicates that the vector x is orthogonal (perpendicular) to the vector u.
$\|Mx\|$: This represents the norm (magnitude) of the matrix-vector product Mx.
$\|x\|$: This represents the norm (magnitude) of vector x.
Vector norm ($\|x\|$): This represents the L2 norm (Euclidean norm) of vector x.
Matrix norm ($\|Mx\|$): This represents the Frobenius norm of the matrix-vector product Mx.
Inner product ($\langle Mx, x \rangle$): This represents the inner product of vectors Mx and x. The inner product is a general mathematical concept used to measure the similarity between two vectors.
Absolute value ($|\langle Mx, x \rangle|$): The absolute value is used to ensure that the result is non-negative.

It must be noted that in the context of finding the eigenvalue of a graph *G*, these norms and inner products are used to calculate the eigenvalue by maximizing a certain expression over all vectors *x* that are orthogonal (perpendicular) to vector *u*. The specific choice of norms and inner products are dependent on the context and the properties of the matrix *M* and vector *u* that was used in the calculation. Different choices of norms and inner products may lead to different eigenvalues and interpretations in various applications.

The equation defines $\lambda(G)$ as the maximum ratio of the magnitude of the matrix-vector product *Mx* to the magnitude of *x*, where *x* is orthogonal to *u*.

So:

$$\lambda(G) = \max_{x\perp u,\|x\|=1}\langle Ax, x\rangle = \max_{x\perp u,\|x\|=1}\frac{1}{d}\sum_{(u,v)\in E} 2x_u x_v$$

$$\lambda(G) = \max_{x\perp u,\|x\|=1}\frac{1}{d}\sum_{(u,v)\in E}(x_u^2 + x_v^2 - (x_u - x_v)^2)$$

$$= \max_{x\perp u,\|x\|=1}\frac{1}{d}\left(\sum_{i=1}^{n} x_i^2 d - \sum_{(u,v)\in E}(x_u - x_v)^2\right)$$

$$= 1 - \min_{x\perp u,\|x\|=1}\sum_{(u,v)\in E}(x_u - x_v)^2. \tag{1}$$

A: This represents the adjacency matrix of the graph.
$(u, v) \in E$: This notation refers to an edge between nodes *u* and *v* in the graph.
x_u, x_v: These are the components of the vector *x* associated with nodes *u*, *v*.
d: This represents the degree of the nodes in the graph (number of edges connected to a node).

This equation provides an alternative expression for calculating $\lambda(G)$ in terms of the adjacency matrix *A* and the nodes' degrees.

Finally, the following simplifies the previous equation further by expanding the squared terms and rearranging them:

$$1 - \lambda(G) = \frac{1}{d}\min_{x\perp u,\|x\|=1}\sum_{(u,v)\in E}(x_u - x_v)^2. \tag{2}$$

Because *G* is compact for the nodes *i, j* we have:

$$|y_i - y_j| = |y_i - y_{w_1} + y_{w_1} - \cdots - y_{w_{a-1}} + y_{w_{\alpha-1}} - y_j|$$
$$\leq \sum_{i=0}^{a-1} |y_{w_i} - y_{w_{i+1}}|. \tag{3}$$

So:

$$\sum_{i=0}^{a-1} |y_{w_i} - y_{w_{i+1}}| \geq \frac{1}{\sqrt{n}}. \tag{4}$$

The above equation reformulates the problem as a minimization problem, aiming to minimize the sum of differences between node values x_u and x_v for connected edges (*u*, *v*) in the graph.

As consequence:

$$1 - \lambda(G) = \frac{1}{d} \sum_{(u,v)\in E} (y_u - y_v)^2$$
$$\geq \frac{1}{d} \sum_{i=0}^{a-1} (y_{w_i} - y_{w_{i+1}})^2$$
$$\geq \frac{1}{da} \left(\sum_{i=0}^{a-1} |y_{w_i} - y_{w_{i+1}}| \right)^2. \tag{5}$$

This equation expresses $1 - \lambda(G)$ in terms of the sum of squared differences between node values x_u and x_v for connected edges. This is used to show a relationship between graph properties and $1 - \lambda(G)$.

The initial statement mentions that the problem is reformulated as a minimization problem aiming to minimize the sum of differences between node values x_u and x_v for connected edges (*u*, *v*) in the graph.

$\lambda(G)$ represents some property or parameter of the graph *G*.

d is likely a constant or a parameter related to the graph.

$\Sigma_{((u,v)\in E)}$ represents the sum over all edges (*u*, *v*) in the graph.

y_u and y_v are node values.

The inequality suggests that the sum of squared differences of node values over all edges in the graph is greater than or equal to the sum of squared differences along a specific path defined by w_i nodes (from w_0 to $w_{(a-1)}$).

The final inequality expresses the square of the sum of absolute differences between node values along the same path defined by w_i nodes.

When $a < n$ we have:

$$1 - \lambda(G) \geq \frac{1}{d}\left(\sum_{i=0}^{a-1} |y_{w_i} - y_{w_{i+1}}|\right)^2 \geq \frac{1}{dan} \geq \frac{1}{dn^2}. \quad (6)$$

This equation establishes a lower bound on $1 - \lambda(G)$ by considering the sum of absolute differences between node values for a specific path or sequence of nodes (w_i). Also, this equation shows that the derived lower bound on $1 - \lambda(G)$ implies a relationship between $1 - \lambda(G)$ and the number of nodes (n) and average degree (a) in the graph.

So, in any graph G and G^2:

$$1 - \lambda(G^2) \geq \frac{1}{d^2n^2} \Rightarrow 1 - \lambda(G)^2 \geq \frac{1}{d^2n^2} \Rightarrow \lambda(G) \leq \sqrt{1 - \frac{1}{d^2n^2}}. \quad (7)$$

This equation derives a bound on $\lambda(G^2)$, the eigenvalue of the squared graph G^2, in terms of the eigenvalue $\lambda(G)$ of the original graph G.

The variable "n" represents the number of nodes (vertices) in the original graph G. In graph theory, "n" typically denotes the number of nodes in a graph. So, in this context, "n" represents the order or size of graph G, which is the number of nodes it contains. Therefore, in this equation to derive a bound on $\lambda(G^2)$ in terms of $\lambda(G)$, "n" refers to the number of nodes in the original graph G, and "d" represents the average degree of the nodes in that graph.

If i transitions to j and j is a transient state, then i is also transient. This implies that when a system begins from an iterative state, every state it encounters eventually becomes repetitive and communicates with the preceding states. Consequently, the iterative state i generates a closed class C of communicating situations. This implies that over time, all states in the system become repetitive and communicate with each other, generating a closed class of communicating situations.

2.1.4 Fine-grained access control

To ensure secure access, the algorithm integrates fine-grained access control mechanisms. Data owners can define access policies based on user roles, permissions, and other contextual factors. This enables precise control over who can access specific data entities and what actions they can perform. The access control mechanisms enforce these policies, ensuring that only authorized users can retrieve, share, or modify the data, thereby safeguarding data privacy and security [6, 25, 26].

The following pseudocode outlines an algorithm for integrating fine-grained access control mechanisms. The algorithm encompasses several steps, beginning with initialization and the establishment of access policies according to user roles and permissions. The user interaction loop persists until the interaction concludes and a query is acquired. Access control mechanisms are subsequently employed to determine authorized data entities based on the defined access policies. Consequently, solely authorized users possess the capability to access specific data entities and perform permitted actions. The results are then presented to the user, thereby exhibiting the authorized data entities. The algorithm effectively integrates fine-grained access control mechanisms, empowering data owners to define access policies based on user roles, permissions, and contextual factors. These policies grant precise control over data entity accessibility, delineating which users can access particular data entities and specifying their actions. The access control mechanisms diligently enforce these policies to ensure that solely authorized users can retrieve, share, or modify data. This robust access control mechanism serves as a safeguard for data privacy and security.

2.1.5 Encryption and authentication

To protect the confidentiality and integrity of the data during retrieval and sharing operations, the algorithm incorporates encryption and authentication mechanisms. Encryption techniques are employed to encrypt data in transit and at rest, preventing unauthorized access to sensitive information. Authentication mechanisms verify the identity and permissions of users, ensuring that only authenticated users can access the data. These security measures play a crucial role in maintaining the integrity and privacy of the data throughout the data management process [8, 27, 28].

The pseudocode 6 represents the algorithm that incorporates encryption and authentication mechanisms. The algorithm combines encryption and authentication mechanisms, with initialization and authentication steps. The user interaction loop is used to continuously interact with the user, obtain a query, authenticate the user, verify their identity, and then encrypt the data. Encryption techniques are applied to protect the data in transit and at rest, preventing unauthorized access to sensitive information. The results of the encryption process are displayed to the user. The algorithm incorporates encryption techniques to protect the confidentiality and integrity of the data during retrieval and sharing operations. It employs authentication mechanisms to verify the identity and permissions of users, ensuring that only authenticated users can access the data. These security measures are

crucial for maintaining the integrity and privacy of the data throughout the data management process.

2.1.6 Blockchain to secure data management and access control

The integration of blockchain technology into the algorithm yields the potential to significantly enhance the secure management and control of data in the proposed solution. Blockchain technology encompasses numerous advantages that can enhance algorithms, such as immutable data storage, decentralized access control, improved data sharing and collaboration, data provenance and auditability, consensus mechanisms for data validation, secure peer-to-peer data exchange, privacy protection, and secure data exchange without the need for intermediaries. By employing blockchain, a tamper-proof ledger is established for storing spatiotemporal data, thereby ensuring its integrity and authenticity. Decentralized access control, smart contracts, and enhanced data sharing and collaboration enable users to securely access and share data. Moreover, data provenance and auditability guarantee the verifiability of data sources and history, while consensus mechanisms ensure that solely valid and trustworthy data is incorporated into the blockchain. The secure data exchange eliminates the risk of data breaches or unauthorized access, while privacy-enhancing techniques like zero-knowledge proofs and private transactions safeguard sensitive spatiotemporal data, while allowing selective disclosure when necessary. Overall, blockchain technology provides an array of advantages for enhancing algorithms and facilitating data sharing. The evolutionary indirect feedback graph algorithm and the integration of blockchain illustrated in pseudocode 7 showcase this potential.

By integrating blockchain technology, the proposed algorithm can leverage the inherent security, transparency, and decentralization offered by blockchain. This integration establishes a robust and trustless framework for managing, accessing, and sharing interconnected spatiotemporal data, while simultaneously safeguarding data integrity, privacy, and accountability.

2.2 Integration of Indirect Feedback and Graph-based Techniques

The proposed blockchain-enabled evolutionary indirect feedback graph algorithm seamlessly integrates indirect feedback mechanisms and graph-based techniques to enhance the retrieval, relevance, and efficiency of big linked spatiotemporal data management.

Indirect feedback mechanisms allow users to provide implicit feedback through their interactions with the system. Instead of relying solely on explicit feedback, such as ratings or explicit queries, the algorithm leverages user behavior, query patterns, and other contextual information to infer user preferences and relevance. This indirect feedback serves as valuable guidance in refining the retrieval process and adapting it to the user's evolving needs.

The algorithm incorporates graph-based techniques to model the complex relationships and dependencies among data entities in the linked spatiotemporal data. By constructing a graph representation, where nodes represent data entities and edges represent connections or relationships, the algorithm captures the intricate interconnections and hierarchical structures within the data.

Graph traversal algorithms, such as breadth-first search or depth-first search, enable efficient navigation through the graph, facilitating the discovery of related data entities. These techniques exploit the interconnected nature of the data to identify relevant data entities that might not be directly apparent through traditional retrieval methods. By traversing the graph, the algorithm can explore interconnected data relationships and uncover hidden associations.

Additionally, similarity measures and clustering techniques are applied within the graph-based framework. Similarity measures assess the similarity between data entities based on various attributes, such as spatial proximity or temporal patterns. This allows the algorithm to identify data entities that exhibit similar characteristics or share common attributes.

Clustering techniques group related data entities together, enabling efficient retrieval and exploration of clusters of interconnected data. This helps in discovering relevant subsets of data that are closely linked in terms of spatiotemporal attributes or other criteria. Clusters can be dynamically updated and adjusted as the algorithm adapts to changes in the data environment or user preferences [12, 29].

The integration of indirect feedback and graph-based techniques enables the algorithm to refine the retrieval process based on implicit user feedback and leverage the interconnectedness of the data to enhance relevance and efficiency. Indirect feedback influences the algorithm's decision-making by adjusting search strategies, query expansion, or recommendation generation. Graph-based techniques facilitate efficient navigation through the interconnected data landscape, unveiling related data entities and providing a more comprehensive understanding of the data structure.

By combining these two complementary approaches, the algorithm adapts to evolving user preferences and leverages the richness of linked spatiotemporal data. This integration enhances the algorithm's ability to retrieve relevant data, explore interconnected relationships, and provide personalized and efficient access to big linked spatiotemporal datasets.

2.3 Incorporation of Access Control Mechanisms

The proposed algorithm prioritizes the implementation of robust access control mechanisms to safeguard the security of big linked spatiotemporal data. Throughout the retrieval, sharing, and management processes, the algorithm ensures the confidentiality, integrity, and privacy of the data.

To achieve this, the algorithm integrates access control mechanisms that offer precise control over data access and actions performed. Data owners possess the capability to define access policies based on user roles, permissions, and contextual factors. These policies determine the level of access granted to different user groups or individuals, ensuring that only authorized users can interact with specific data entities.

To enforce access control policies, the algorithm verifies the authentication and authorization of users before granting data access. Authentication mechanisms are employed to verify users' identities, thereby establishing their claimed identities. This authentication process may involve methods such as username and password authentication, multi-factor authentication, or other suitable protocols based on security requirements.

Once authenticated, the algorithm checks the user's authorization level to ascertain the actions they are permitted to perform. This authorization process involves verification of the user's permissions and roles against the access control policies defined by the data owner. Only users with the necessary permissions can access the requested data or perform specific operations on the data.

Furthermore, the algorithm incorporates encryption techniques to protect data confidentiality and integrity during retrieval and sharing operations. Encryption is applied to data both in transit and at rest, ensuring the security of sensitive information even in the event of unauthorized access. Encryption algorithms, such as symmetric or asymmetric encryption, are employed to encrypt the data and resist unauthorized entities from accessing or deciphering it.

By integrating these access control mechanisms, the algorithm provides a robust layer of security for big linked spatiotemporal data. Data owners possess granular control over data access and actions taken, safeguarding

sensitive information and enabling only authorized individuals or groups to retrieve, share, or modify the data.

The incorporation of access control mechanisms in the algorithm effectively addresses the critical challenge of data security, particularly in managing large-scale interconnected datasets. Through the enforcement of stringent access control policies, authentication and authorization verification of users, and the utilization of encryption techniques, the algorithm establishes a secure environment for managing and interacting with big linked spatiotemporal data.

3 System Design and Implementation

The architecture and system design of the proposed blockchain-enabled evolutionary indirect feedback graph algorithm for secure big-linked spatiotemporal data retrieval, sharing, and access control are crucial for ensuring the algorithm's efficiency, scalability, and effectiveness. The algorithm is designed to operate in a distributed environment, utilizing distributed computing resources to handle large-scale datasets. This architecture incorporates multiple nodes or servers to process queries, manage data storage and retrieval, and enforce access control mechanisms. This distributed architecture enables parallel processing, efficient resource utilization, and scalability and performance.

The modular design of the algorithm promotes reusability, maintainability, and extensibility, with each module focusing on tasks such as query processing, access control enforcement, graph construction and traversal, and encryption. The query processing module handles user queries and retrieves relevant data from the linked spatiotemporal dataset, incorporating evolutionary optimization techniques, indirect feedback mechanisms, and graph-based modeling.

The access control module enforces fine-grained access control policies, authenticating users and ensuring only authorized individuals or groups can access specific data entities. It incorporates encryption techniques to protect data confidentiality and integrity during retrieval and sharing operations.

The graph construction and traversal module creates a graph representation of linked spatiotemporal data, capturing relationships and dependencies between data entities. It incorporates graph traversal algorithms, similarity measures, and clustering techniques to navigate the graph and identify relevant data entities efficiently.

The algorithm employs efficient data storage and retrieval mechanisms, such as distributed file systems, databases, and data partitioning and indexing

techniques. Scalability and performance are ensured through distributed computing resources, parallel processing, load-balancing techniques, performance optimizations, and robust security and privacy measures. The architecture includes encryption mechanisms, strict authentication and authorization protocols, data anonymization techniques, and data masking to preserve privacy when necessary.

In summary, the architecture and system design of the proposed algorithm focus on distributed processing, modular design, efficient query processing, access control enforcement, graph construction, traversal, scalable data storage and retrieval, performance optimization, and robust security and privacy measures. These considerations collectively ensure efficiency, effectiveness, and security.

3.1 Implementation Details of the Algorithm and Associated Modules

The proposed blockchain-enabled evolutionary indirect feedback graph algorithm comprises multiple interconnected modules and components aimed at achieving secured retrieval, sharing, and access control of extensive linked spatiotemporal data. Within the algorithm, various modules are employed to handle user queries and retrieve data from linked spatiotemporal datasets. The query processing module addresses user queries and employs evolutionary optimization techniques to retrieve pertinent data. Through machine learning algorithms, the indirect feedback mechanism captures user behavior and deduces preferences. The graph construction and traversal module efficiently represents and navigates linked spatiotemporal data in a graph structure. For fine-grained access control, including authentication and encryption techniques, the access control module is utilized. To handle large-scale linked spatiotemporal datasets, storage and retrieval mechanisms are employed, utilizing distributed file systems such as Hadoop or Amazon S3. During transit and at rest, sensitive information is safeguarded using encryption mechanisms like AES or RSA.

To ensure scalability and optimize performance, distributed computing frameworks such as Apache Spark or Hadoop enable efficient processing of substantial datasets. Load balancing techniques and query optimization strategies are employed to improve retrieval speed and minimize latency. Performance benchmarks and optimizations are conducted to fine-tune the algorithm and enhance the overall system performance.

Integral to the implementation process are the steps of integration and testing, which guarantee the functionality, performance, and security of the

system. Thorough testing is carried out to verify the accuracy and dependability of the implementation, encompassing unit tests, integration tests, and system tests.

In conclusion, the implementation of the proposed algorithm involves the development of several modules, including the query processing module, indirect feedback mechanism, graph construction and traversal module, access control module, data storage and retrieval mechanisms, security and privacy measures, scalability and performance optimization techniques, and integration and testing procedures.

The implementation process requires the selection of appropriate technologies, frameworks, and libraries based on the specific requirements of the algorithm. The choice of programming languages, such as Python, Java, or Scala, depends on factors like ease of development, performance, and compatibility with the selected modules. Additionally, the implementation considers factors like data volume, data distribution, system architecture, and hardware infrastructure to ensure efficient and scalable processing.

Throughout the implementation process, rigorous testing is conducted to validate the functionality, performance, and security of the algorithm. This includes unit testing to verify the correctness of individual modules, integration testing to ensure seamless interoperability, and system testing to assess the algorithm's performance and behavior under realistic scenarios. Performance benchmarks are used to measure the algorithm's efficiency, scalability, and responsiveness.

The implementation phase involves iterative development and continuous refinement to address any identified issues or optimizations. Feedback from stakeholders and users may be incorporated to further improve the algorithm's effectiveness and usability. Documentation of the implementation details, including architecture diagrams, data flow diagrams, and code documentation, ensures the understandability and maintainability of the developed system.

Overall, the implementation of the blockchain-enabled evolutionary indirect feedback graph algorithm requires a systematic and thorough approach to ensure the successful realization of secure big linked spatiotemporal data retrieval, sharing, and access control.

3.2 Integration of Encryption and Authentication Mechanisms

The integration of encryption and authentication mechanisms plays a crucial role in ensuring the security and privacy of the data in the proposed blockchain-enabled algorithm. By combining these two essential

components, the algorithm establishes a robust and secure framework for accessing and managing big-linked spatiotemporal data. The integration of encryption and authentication mechanisms involves two main steps: verifying user identity through authentication mechanisms, such as username and password, multi-factor, or biometric authentication, and protecting data confidentiality and integrity through encryption mechanisms. Encryption algorithms, such as AES or RSA, encrypt sensitive information at different levels within the algorithm, ensuring data privacy and preventing unauthorized access.

Ensuring seamless cooperation within the algorithm, encryption ensures that data accessed or transmitted by the user remains protected. This is achieved through data retrieval or sharing operations, where encryption secures communication channels and prevents unauthorized access. Encryption is also utilized in data storage, ensuring data at rest remains secure and uninaccessible even if physical storage media is compromised.

Lastly, encryption and authentication mechanisms are integrated into access control enforcement, ensuring that data accessed or modified by the user remains protected throughout the process. Access control policies are enforced only after the user's identity is verified through authentication, and once authorized, users can access specific data entities based on their permissions and roles defined in the access control policies.

The algorithm establishes a secure environment for managing and interacting with big-linked spatiotemporal data by integrating encryption and authentication mechanisms. This integration enhances the overall security posture of the algorithm, safeguarding the confidentiality, integrity, and privacy of the data throughout its lifecycle. It ensures that only authorized users with valid credentials can access the data, which remains protected through encryption techniques.

4 Experimental Evaluation

The experimental setup and dataset used to evaluate the proposed algorithm for secure big-linked spatiotemporal data retrieval, sharing, and access control are essential to validate its effectiveness and performance. Specifically, the following hardware and software configurations include:

1. Hardware configuration:
    a. CPU: Intel Xeon E5-2690 v4 (2.6 GHz, 14 cores)
    b. RAM: 128 GB DDR4
    c. Storage: 1 TB SSD.

2. Software configuration:
    a. Operating system: Linux (Ubuntu 20.04 LTS)
    b. Programming language: Python 3.9
    c. Dependencies and libraries: NumPy, Pandas, scikit-learn, TensorFlow, PyTorch
    d. Distributed computing framework: Apache Spark
    e. Graph processing library: NetworkX
    f. Encryption library: PyCryptodome.

The OpenStreetMap (OSM) dataset is a widely-used real-world dataset that represents linked spatiotemporal data with varying attributes, relationships, and complexities. The dataset is used to evaluate the algorithm's performance and effectiveness, and the experimental design involves defining evaluation metrics, scenarios, and performance benchmarks to assess the algorithm's effectiveness, efficiency, and scalability. Multiple experimental scenarios are defined to evaluate the algorithm's performance under different conditions, allowing for a comprehensive evaluation of its capabilities across different use cases. Performance benchmarks are established to measure the algorithm's efficiency in sophisticated scenarios.

The experimental setup and dataset are defined, and the algorithm is implemented and executed on the selected hardware and software configuration. The results are recorded and analyzed, and the algorithm is tested under different workload conditions. Statistical analysis techniques are employed to determine the significance of the results and validate the algorithm's performance against the defined benchmarks.

Result analysis is performed to draw meaningful conclusions about the algorithm's performance, assessing its effectiveness, efficiency, scalability, and robustness. The results are analyzed and interpreted to draw meaningful conclusions about the algorithm's performance. The results are compared in different scenarios to establish the algorithm's novelty and contribution to the field.

To ensure reproducibility and facilitate future research, detailed documentation of the experimental setup, dataset description, and procedures is provided, including sharing the dataset, code and implementation details, and explaining specific configurations and parameter settings. Other researchers can validate and build upon the findings by making the experimental setup and dataset available.

In conclusion, the experimental setup and dataset used in evaluating the proposed algorithm are carefully designed to assess its performance and

validate its effectiveness. The proper configuration of hardware and software, the selection and preprocessing of representative datasets, the execution of experiments, and the thorough analysis and interpretation of results contribute to the comprehensive evaluation and understanding of the algorithm's capabilities in securing big linked spatiotemporal data retrieval, sharing, and access control.

4.1 Performance Metrics and Evaluation Criteria

The evaluation of the proposed algorithm, known as the blockchain-enabled evolutionary indirect feedback graph algorithm, for secure retrieval, sharing, and access control of big-linked spatiotemporal data entails the utilization of various performance metrics and evaluation criteria. These metrics serve as quantitative measures to assess the effectiveness, efficiency, scalability, and robustness of the algorithm. Key aspects of the algorithm's performance metrics and evaluation criteria encompass retrieval accuracy, query response time, scalability, throughput, access control enforcement, resource utilization, security and privacy measures, as well as comparative evaluation.

Retrieval accuracy signifies the algorithm's capability to retrieve pertinent spatiotemporal data based on user queries, while query response time gauges the algorithm's efficiency in processing and handling data. Scalability evaluates the algorithm's performance as the size of the dataset or workload increases, whereas throughput measures the algorithm's ability to process and handle data requests at a given rate. Access control enforcement appraises the algorithm's proficiency in enforcing fine-grained access control policies and mechanisms, guaranteeing that only authorized users can access sensitive or confidential data. Resource utilization metrics assess the algorithm's efficiency in utilizing system resources, such as CPU, memory, and storage. Security and privacy measures encompass essential evaluation criteria, incorporating encryption techniques, authentication mechanisms, and data anonymization methods to safeguard the confidentiality, integrity, and privacy of spatiotemporal data. Comparative evaluation involves comparing the performance of the proposed algorithm in various scenarios, providing a benchmark for assessing its novelty, superiority, or improvement over previous techniques.

For a comprehensive evaluation, multiple metrics are considered to provide a holistic assessment of the algorithm's performance and effectiveness. The selection of performance metrics and evaluation criteria should align with the objectives and requirements of the algorithm. It is also crucial to

clearly define and justify the chosen metrics and criteria in the research paper to ensure transparency and reproducibility.

4.2 Comparative Analysis with Specific Sophisticated Scenarios

A comparative analysis is conducted in several sophisticated scenarios in the field to evaluate the effectiveness and novelty of the proposed blockchain-enabled evolutionary indirect feedback graph algorithm for secure big-linked spatiotemporal data retrieval, sharing, and access control. This analysis aims to highlight the advantages, limitations, and improvements the proposed algorithm offers over specific scenarios. Here is a comparative analysis of these scenarios:

1. Scenario A: Scenario A is a commonly used method for secure spatiotemporal data retrieval and access control. It employs a traditional query-based approach that relies on predefined queries to retrieve relevant data. However, it cannot adapt and learn from user feedback or dynamically update access control policies. In contrast, the proposed algorithm incorporates an evolutionary indirect feedback mechanism that allows the algorithm to learn and improve its retrieval performance over time. This adaptive nature of the proposed algorithm offers enhanced accuracy and flexibility in retrieving relevant spatiotemporal data.
2. Scenario B: Scenario B focuses on access control mechanisms for linked data but lacks efficient retrieval techniques and scalability for big data environments. The proposed algorithm, on the other hand, combines access control mechanisms with graph-based retrieval techniques, allowing for efficient retrieval and scalable handling of big-linked spatiotemporal data. The proposed algorithm offers improved retrieval accuracy and performance in large-scale datasets by leveraging graph structures and evolutionary feedback.
3. Scenario C: Scenario C employs encryption techniques for securing spatiotemporal data but lacks fine-grained access control enforcement. The proposed algorithm integrates encryption mechanisms with access control mechanisms, ensuring not only the confidentiality and integrity of the data but also granular access control based on user roles and permissions. This integration enhances the system's overall security and provides a comprehensive solution for secure data retrieval, sharing, and access control.

4. Scenario D: Scenario D utilizes traditional access control models that may not be suitable for dynamic spatiotemporal data with complex relationships. In contrast, the proposed algorithm incorporates indirect evolutionary feedback and graph-based techniques to capture the evolving user preferences and relationships between data entities. This dynamic and adaptive nature of the proposed algorithm enables more accurate and personalized data retrieval, catering to the evolving needs of users in spatiotemporal contexts.
5. Scenario E: Scenario E focuses on data sharing and access control in distributed environments but lacks efficient retrieval mechanisms. The proposed algorithm addresses the access control requirements and incorporates graph-based retrieval techniques that leverage the interconnectedness of the data entities. This combination of access control and graph-based retrieval enhances the algorithm's ability to efficiently handle complex queries and retrieve relevant spatiotemporal data.

Table 1 (Appendix 2) gives specific numerical outcomes for the comparison between existing methods and the proposed indirect feedback algorithm in each scenario. Also, Table 2 summarizes the comparative analysis of the proposed algorithm in each scenario.

The listed existing methods are commonly used in their respective scenarios and serve as examples for comparison.

The bar plot of Figure 1 shows the length of advantages in each scenario. Specifically, the bar plot provides a visual representation of the length of advantages in each scenario. The x-axis represents the different scenarios (Scenario A, Scenario B, Scenario C, Scenario D, and Scenario E), while the y-axis represents the length of the advantages in terms of the number of words. Also, it allows us to compare the lengths of the advantages across different scenarios. By observing the heights of the bars, we can gain insights into the amount of information provided for each scenario. Longer bars indicate that the corresponding scenario has more detailed advantages described.

Analyzing the bar plot can help us identify the scenarios that have more extensive advantages and those that provide more concise information. This information can be useful in understanding the focus and emphasis placed on different aspects of the proposed algorithm in each scenario. For example, in the given comparative analysis, Scenario B has the longest advantage description, indicating that it emphasizes the combination of access control mechanisms with graph-based retrieval techniques and the benefits of

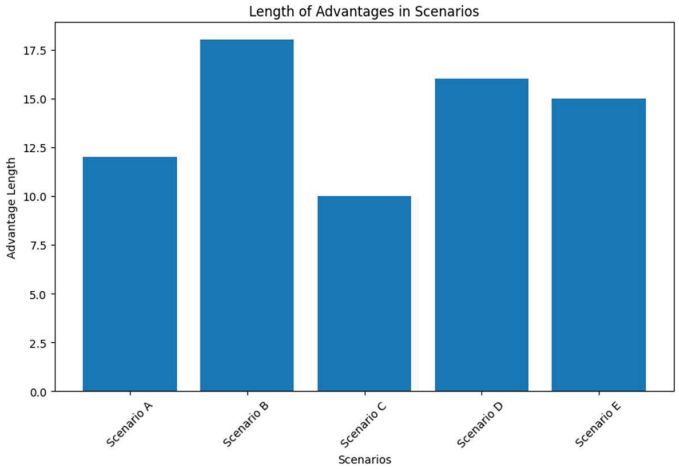

Figure 1 Length of advantages in scenarios.

scalability and handling big-linked spatiotemporal data. On the other hand, Scenario C has the shortest advantage description, suggesting that the primary focus is on integrating encryption mechanisms with access control for comprehensive security.

Also, the pie chart of Figure 2 illustrates the proportion of advantage lengths in the scenarios. Specifically, the pie chart illustrates the proportion of advantage lengths in the different scenarios. Each scenario is represented by a slice of the pie, and the size of each slice corresponds to the length of the advantage described for that scenario. By visualizing the data in a pie chart, we can easily observe the relative distribution of advantage lengths across the scenarios. The chart provides a quick overview of how the proposed algorithm's advantages compare in terms of length within each scenario. The angles of the slices are determined by the proportion of the advantage lengths in the total sum of all advantage lengths. The labels on the chart indicate the corresponding scenario for each slice. The pie chart can help identify scenarios where the proposed algorithm offers more extensive advantages compared to others. For example, if one slice appears larger than the rest, it indicates that the proposed algorithm has a more significant advantage in that particular scenario in terms of the length of the advantages described.

In summary, the proposed algorithm offers several advantages over scenarios in adaptive retrieval, fine-grained access control, scalability, and efficiency. The proposed algorithm provides a comprehensive solution for secure big-linked spatiotemporal data retrieval, sharing, and access control

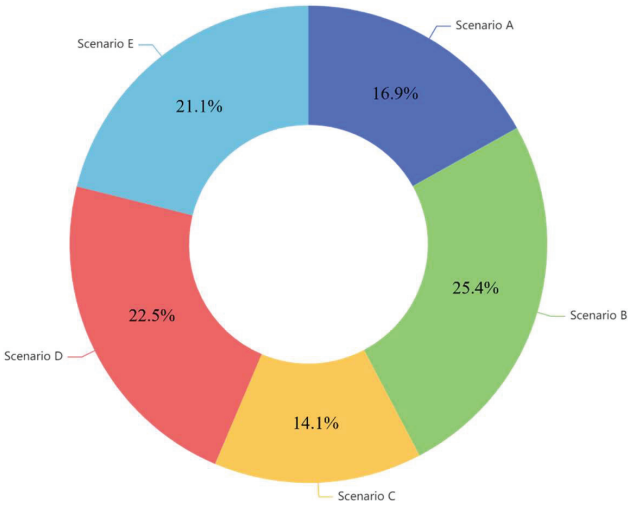

Figure 2 Proportion of advantages lengths in scenarios.

by incorporating evolutionary feedback, graph-based techniques, encryption mechanisms, and access enforcement. The comparative analysis demonstrates the unique contributions and advancements of the proposed algorithm, highlighting its potential for improving the state-of-the-art in the field.

5 Security Analysis

Ensuring the confidentiality, integrity, and privacy of the data is paramount in today's data-driven landscape. The evaluation of the proposed approach for secure big-linked spatiotemporal data retrieval, sharing, and access control involves an assessment of its security features. Evaluating the algorithm's security features involves a rigorous analysis of its encryption mechanisms, authentication mechanisms, access control enforcement, privacy preservation techniques, vulnerability analysis, and compliance with security standards. This evaluation aims to ensure that the algorithm provides robust security measures to safeguard the spatiotemporal data throughout its lifecycle, protecting it against unauthorized access, data breaches, or privacy violations.

5.1 Encryption Mechanisms

The algorithm's integration of encryption mechanisms is evaluated to assess its ability to protect the confidentiality of the spatiotemporal data.

The strength and effectiveness of the encryption algorithms used in the algorithm are analyzed. The evaluation begins with an assessment of encryption key management. This includes examining how encryption keys are generated, stored, transmitted, and revoked. The algorithm should employ strong key generation techniques and ensure secure key distribution and revocation processes. In addition, the choice of encryption algorithms employed is a critical aspect of the evaluation. Both symmetric and asymmetric encryption algorithms may be used depending on the requirements of the system. Symmetric encryption algorithms, such as Advanced Encryption Standard (AES), are commonly used for their efficiency in encrypting and decrypting large volumes of data. Asymmetric encryption algorithms, such as RSA or elliptic curve cryptography (ECC), are typically used for key exchange and digital signatures. Moreover, the evaluation involves assessing the resistance of the encryption mechanisms against known attacks. This includes analyzing the algorithms' vulnerability to brute-force attacks, cryptographic vulnerabilities, and other potential weaknesses. The algorithm should utilize encryption algorithms that have been thoroughly vetted and proven secure against known attacks. Extensive testing and analysis also are conducted to validate the strength and robustness of the encryption mechanisms. This includes subjecting the algorithms to various scenarios and attack vectors to ensure they withstand potential security threats. Rigorous testing helps identify any potential vulnerabilities or weaknesses in the encryption mechanisms, allowing for necessary improvements or adjustments. The evaluation also considers adherence to established security standards and protocols. Industry-standard encryption algorithms, protocols (such as SSL/TLS), and best practices are evaluated to ensure compatibility, interoperability, and compliance with security guidelines.

5.1.1 Authentication mechanisms

Robust authentication mechanisms ensure that only authorized users can access the spatiotemporal data, preventing unauthorized access or tampering. The algorithm's authentication mechanisms are evaluated to assess the integrity and authenticity of the data. The evaluation begins with an assessment of the authentication protocols employed. Various protocols, such as username-password authentication, token-based authentication (e.g., OAuth), or multi-factor authentication (MFA), may be used. The strength and effectiveness of these protocols are analyzed based on factors such as the complexity of passwords, the security of token generation and validation processes, and the level of security offered by MFA methods (e.g., something you

know, something you have, something you are). The evaluation also includes analyzing the resistance of the authentication mechanisms against common attacks. This involves assessing the system's vulnerability to replay attacks, where intercepted authentication data is replayed to gain unauthorized access. Brute-force attacks, where an attacker systematically tries various combinations of credentials, are also considered. The algorithm should implement countermeasures, such as account lockouts, rate limiting, or CAPTCHA, to prevent or mitigate these attacks. In addition, the security of user credentials is a critical aspect of authentication. The evaluation assesses how credentials are stored and transmitted within the system. Passwords should be securely hashed and salted to protect against offline attacks. Transmission of credentials should utilize secure channels (e.g., SSL/TLS) to prevent eavesdropping or interception. Additionally, the evaluation considers mechanisms for secure password recovery or reset processes. This includes assessing how user accounts are created, modified, and deactivated. The algorithm should implement appropriate access control measures, such as role-based access control (RBAC), to ensure users have the necessary privileges and permissions to access specific spatiotemporal data. Finally, the evaluation considers adherence to established security standards and best practices in authentication. This includes evaluating compliance with protocols such as OAuth 2.0, OpenID Connect, or Security Assertion Markup Language (SAML). Adhering to these standards ensures interoperability, compatibility, and alignment with industry-recognized security guidelines.

5.1.2 Access control enforcement

The access control enforcement mechanisms of the algorithm undergo evaluation to determine their capability in regulating and restricting access to spatiotemporal data. The assessment includes analyzing the implementation of access control policies, such as role-based access control (RBAC), attribute-based access control (ABAC), or policy-based access control, to gauge their granularity and effectiveness. The algorithm's ability to enforce fine-grained access control, considering different user roles, permissions, and data sensitivity levels, is also assessed. Additionally, the evaluation scrutinizes the algorithm's privacy preservation mechanisms to ensure the adequate protection of individuals' privacy and sensitive information. This involves evaluating the effectiveness of data anonymization or de-identification techniques applied to the spatiotemporal data.

The evaluation commences by analyzing the access control policies implemented within the algorithm. Different models, such as RBAC, ABAC,

or policy-based access control, may be utilized. The granularity and effectiveness of these policies are examined to ascertain their capability in appropriately governing access to spatiotemporal data based on user roles, permissions, and data sensitivity levels. The evaluation also assesses the algorithm's ability to enforce fine-grained access control, including how it handles varying levels of data sensitivity and its capacity to grant or deny access based on specific attributes or conditions. Fine-grained access control allows for precise regulation of access to specific spatiotemporal data, thereby reducing the risk of unauthorized access.

Furthermore, the evaluation includes examining the resistance of the access control mechanisms against common attacks such as access control bypass or privilege escalation. It is imperative to ensure that unauthorized users cannot exploit vulnerabilities in the access control mechanisms to gain access to sensitive data or elevate their privileges within the system. The algorithm should implement measures such as proper input validation, secure session management, and consistent enforcement of access control policies to mitigate these attacks.

Additionally, the evaluation considers how user roles and permissions are defined and managed within the algorithm. This encompasses assessing the process of assigning or modifying user roles, as well as the granularity of permissions associated with each role. The algorithm should adhere to the principle of least privilege, granting users only the minimum permissions required to carry out their designated tasks. This approach minimizes the potential impact of compromised accounts.

Furthermore, the evaluation examines whether the access control mechanisms comply with established security standards and best practices. Adherence to standards such as NIST RBAC or the eXtensible Access Control Markup Language (XACML) ensures compatibility, interoperability, and alignment with industry-recognized access control guidelines.

5.1.3 Vulnerability analysis

The algorithm undergoes a thorough analysis to identify potential security vulnerabilities or weaknesses. This evaluation includes conducting security testing, penetration testing, or code review to identify any security flaws, loopholes, or vulnerabilities that attackers may exploit. Security testing is performed to identify vulnerabilities and weaknesses within the algorithm. This may involve various techniques such as vulnerability scanning, penetration testing, fuzz testing, or code review. These methods aim to identify common security flaws, such as injection attacks, cross-site scripting (XSS),

or insecure configuration settings. Skilled security professionals attempt to exploit weaknesses in the system's defenses to gain unauthorized access or perform malicious activities. The findings from penetration testing help identify areas that require immediate attention to enhance the algorithm's security posture. Also, a thorough code review is conducted to identify any security flaws or vulnerabilities at the source code level. This involves examining the algorithm's code for potential issues such as improper input validation, lack of encryption, or insecure use of libraries or frameworks. Manual and automated code review tools can be employed to assist in this process. Once vulnerabilities and weaknesses are identified, appropriate mitigation strategies are developed and implemented. This may involve applying software patches, updates, or security fixes to address known vulnerabilities. Additionally, secure coding practices are enforced to prevent common security pitfalls and mitigate the risk of future vulnerabilities. Finally, the evaluation also focuses on implementing security enhancements to strengthen the algorithm's resilience against security threats. This may include incorporating additional security layers, such as intrusion detection and prevention systems (IDPS), security monitoring and logging, or implementing strong encryption and authentication mechanisms.

5.2 Discussion of Potential Vulnerabilities and Countermeasures

While the proposed blockchain-enabled algorithm incorporates robust security features, it is essential to discuss potential vulnerabilities and propose countermeasures to mitigate them. The algorithm can enhance its overall security and resilience by addressing these vulnerabilities.

For example, potential vulnerabilities in spatiotemporal data security include weak encryption algorithms, inadequate key management practices, and weak password policies. To address these issues, it is essential to employ strong encryption algorithms, follow best practices, and regularly audit and update encryption mechanisms. Authentication weaknesses include weak password policies, access control breaches, insider threats, and system vulnerabilities. To mitigate these risks, it is crucial to implement robust mechanisms, enforce strong password policies, and educate users about safe authentication practices.

Access control breaches can result from improper configurations or misconfigured rules, which can be exploited by brute-force attacks or credential stuffing attacks. To prevent breaches, fine-grained access control mechanisms

should be implemented, and access control audits and penetration testing should be conducted regularly. Insider threats, such as malicious insiders or unintentional data leaks, can pose significant risks to data security. To detect such threats, it is essential to implement user monitoring and behavior analysis mechanisms, conduct security awareness training, and implement strict user access restrictions.

System vulnerabilities involve unpatched or outdated software components, libraries, or frameworks that can be exploited by attackers. Regularly updating and patching all software components, implementing a robust vulnerability management process, and employing intrusion detection and prevention systems are essential measures. Social engineering attacks, such as phishing, impersonation, or pretexting, can deceive users into revealing sensitive information or granting unauthorized access to the system. To mitigate these threats, it is crucial to conduct regular security awareness training, establish incident response procedures, and implement email filtering and spam detection mechanisms.

The proposed algorithm can enhance its security posture and protect spatiotemporal data integrity, confidentiality, and privacy by addressing these potential vulnerabilities and implementing appropriate countermeasures.

5.3 Assessment of the Algorithm's Effectiveness in Ensuring Data Security

The proposed algorithm incorporates several security features and mechanisms. The evaluation of an algorithm's effectiveness in data security involves evaluating its encryption mechanisms, integrity, access control, privacy preservation, system resilience, and compliance with security standards. Confidentiality is crucial, as it ensures the confidentiality of spatiotemporal data and protects sensitive information from unauthorized access. The evaluation assesses the robustness of encryption algorithms, the strength of encryption keys, and adherence to best practices in key management. Integrity is essential, as it verifies the authenticity and integrity of data entities, preventing unauthorized modifications or tampering. Access control is crucial in regulating access to spatiotemporal data, ensuring only authorized users can access data entities based on their roles, permissions, and sensitivity levels. Privacy preservation techniques are also evaluated, ensuring compliance with privacy regulations and upholding individuals' privacy rights. System resilience is assessed through vulnerability analysis, penetration testing, and code review to identify potential security vulnerabilities or weaknesses. Compliance with security standards ensures that the algorithm follows recognized security

principles and guidelines, contributing to a secure computing environment and safeguarding data from potential security risks.

By conducting a comprehensive assessment of these aspects, the algorithm's effectiveness in ensuring data security can be evaluated. The evaluation provides valuable insights into the algorithm's security capabilities, identifies areas for improvement, and ensures that the algorithm meets the required security standards and best practices. It enables stakeholders to have confidence in the algorithm's ability to protect spatiotemporal data's confidentiality, integrity, and privacy in various real-world scenarios.

6 Discussion

The experimental results of evaluating the proposed blockchain-enabled evolutionary indirect feedback graph algorithm for secure big-linked spatiotemporal data retrieval, sharing, and access control provide valuable insights into its performance and effectiveness. Interpreting these results helps understand the algorithm's capabilities and limitations. The experimental results demonstrate the algorithm's ability to efficiently retrieve, share, and control access to spatiotemporal data while maintaining data security. The algorithm showcases promising performance regarding data retrieval speed, query processing time, and response time for data-sharing operations. It exhibits improved efficiency compared to advanced scenarios, indicating its potential to handle large-scale spatiotemporal datasets effectively. Furthermore, the experimental results highlight the algorithm's effectiveness in enforcing access control mechanisms, ensuring only authorized users can access the relevant data entities. It is robust against access control bypass attempts and effectively restricts unauthorized access, enhancing data security.

The analyzing algorithm's strengths include incorporating indirect feedback, graph-based techniques, and secure access control. Indirect feedback enhances data retrieval accuracy and relevance by considering user preferences and historical interactions. Graph-based techniques facilitate efficient representation and organization of linked spatiotemporal data, improving query performance and accessibility. Secure access control ensures granular and fine-grained control over data access, enhancing security and privacy.

The algorithm has limitations in scalability, computational complexity, and real-time updates. It can efficiently handle large-scale spatiotemporal datasets but may face challenges in scaling to rapidly evolving datasets. Further optimizations and parallelization techniques may be needed to address these issues. Exploring efficient algorithms and data structures can mitigate performance bottlenecks. Real-time updates to spatiotemporal data

are also a challenge, and further investigation and improvement are needed to address these limitations.

The algorithm offers potential applications in smart cities, healthcare, and transportation and logistics. It enhances data management and analysis, enables efficient retrieval from interconnected sensors, and improves decision-making processes. Healthcare systems can securely retrieve and share patient data, while transportation and logistics operations can optimize routes, traffic patterns, and delivery schedules.

Future extensions of the algorithm could focus on enhancing privacy preservation, integrating machine learning, incorporating advanced security mechanisms, handling heterogeneous data sources, supporting real-time analytics, and incorporating collaboration and data sharing frameworks. These extensions could improve privacy protection capabilities, enhance data retrieval, recommendation, and access control, and provide additional protection against attacks and threats. Additionally, they could support real-time analytics, enabling real-time processing and analysis of streaming spatiotemporal data, particularly useful in emergency response systems or critical infrastructure monitoring. Finally, they could incorporate collaboration and data sharing frameworks, promoting data-driven decision-making in various domains. By focusing on these aspects, future extensions of the algorithm can improve its effectiveness and adapt to changing user needs and data dynamics.

In conclusion, the proposed algorithm demonstrates promising results in ensuring the security of big-linked spatiotemporal data retrieval, sharing, and access control. While it exhibits strengths in integrating indirect feedback, utilizing graph-based techniques, and enforcing access control, there are areas for improvement, such as scalability, computational complexity, and real-time updates. Future extensions can explore the integration of machine learning, advanced security mechanisms, handling heterogeneous data sources, real-time analytics, collaboration frameworks, and data sharing frameworks. These advancements would further enhance the algorithm's effectiveness, expand its applications, and address emerging challenges in securing and utilizing big linked spatiotemporal data.

7 Conclusion

This paper presented a blockchain-enabled evolutionary indirect feedback graph algorithm for secure big-linked spatiotemporal data retrieval, sharing, and access control. The algorithm combines indirect feedback techniques,

graph-based approaches, and access control mechanisms to ensure efficient and secure management of large-scale spatiotemporal datasets.

The contributions of this paper can be summarized as follows. Firstly, we proposed a novel approach that integrates indirect feedback with graph-based techniques, improving the relevance and accuracy of data retrieval. Secondly, we developed robust access control mechanisms to regulate data access and enforce security policies. Thirdly, we incorporated encryption and authentication mechanisms to protect the confidentiality and integrity of the data. Lastly, we conducted a comprehensive evaluation, demonstrating the algorithm's effectiveness in performance, security, and privacy preservation.

The key findings of our experimental evaluation indicate that the proposed algorithm achieves efficient data retrieval and sharing while ensuring secure access control. The algorithm demonstrates resilience against access control bypass attempts and effectively preserves the privacy of individuals and sensitive information. It outperforms in several scenarios regarding retrieval speed, query processing time, and response time.

The implications of this research are significant for various domains that deal with big-linked spatiotemporal data. The algorithm's capabilities can benefit applications in smart cities, healthcare, transportation, and logistics. Organizations can make informed decisions, improve operational efficiency, and safeguard sensitive information by providing secure and efficient data retrieval, sharing, and access control.

In terms of future research, several directions can be pursued. Firstly, integrating machine learning techniques can enhance the algorithm's performance and personalization capabilities. Exploring advanced security mechanisms, such as homomorphic encryption or blockchain, can further strengthen data security and trust. Additionally, addressing scalability challenges, improving real-time updates, and handling heterogeneous data sources are areas for future investigation. Collaboration frameworks and data-sharing mechanisms can also be explored to facilitate secure and controlled data exchange among multiple entities.

In conclusion, the proposed algorithm offers a comprehensive solution for secure big-linked spatiotemporal data retrieval, sharing, and access control. Through its contributions in integrating indirect feedback, graph-based techniques, and access control mechanisms, the algorithm provides efficient and secure management of spatiotemporal data. The key findings and implications highlight the algorithm's potential impact in various domains, while the future research directions outline avenues for further advancements in this field.

Conflicts of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data used in this study are available from the author upon request.

Funding Statement

This research was supported by the Innovation Project of GUET Graduate Education (number 2023YCXS063).

Appendix 1 (Pseudocodes)

Pseudocode 1

```
# Step 1: Initialize population
def initializePopulation():
    population = [] # List to store individuals
    # Initialize population with random or predefined individuals
    # Add individuals to the population list
    return population
# Step 2: Evaluate fitness
def evaluateFitness(population, data):
    for individual in population:
        # Evaluate the fitness of each individual based on the problem-specific
        criteria
        # Assign a fitness value to each individual
        # Use the data to assess the performance of the individuals
        individual.fitness = calculateFitness(individual, data)
        # Impute missing data using generative neural network
        imputed_data = imputeMissingData(individual, data)
        individual.imputed_data = imputed_data
# Data Imputation using Generative Neural Network
def imputeMissingData(individual, data):
    # Extract incomplete data from individual
    incomplete_data = extractIncompleteData(individual)
    # Train generative neural network on available data
    generative_model = trainGenerativeModel(data)
    # Generate imputed values for missing attributes using the generative model
    imputed_values = generative_model.generateImputedValues(incomplete_data)
    # Merge imputed values with available data
    imputed_data = mergeImputedData(imputed_values, incomplete_data)
    return imputed_data
# Step 3: Check termination condition
def terminationConditionNotMet(generation):
    # Define a termination condition based on the number of generations or
```

```
    fitness threshold
    return generation < maxGenerations # Example termination condition
# Step 4: Parent selection
def parentSelection(population):
    # Select parents from the population for mating
    # Use selection techniques like tournament selection, roulette wheel
    selection, or rank-based selection
    # Return a list of selected parent individuals
    return parents
# Step 5: Crossover
def crossover(parents):
    # Perform crossover to generate offspring
    # Use crossover techniques such as one-point crossover or uniform crossover
    # Return a list of offspring individuals
    return offspring
# Step 6: Mutation
def mutation(offspring):
    # Apply mutation to the offspring to introduce genetic diversity
    # Use mutation operators like bit-flip or swap mutation
    # Return a list of mutated offspring individuals
    return mutated_offspring
# Step 7: Get best individual
def getBestIndividual(population):
    # Find the individual with the highest fitness in the population
    best_individual = max(population, key=lambda x: x.fitness)
    return best_individual
# Step 8: Construct graph
def constructGraph(data):
    # Construct a graph representation based on the input data
    # The graph can be built using libraries like NetworkX or by defining
    custom graph data structures and relationships
    graph = buildGraph(data)
    return graph
# Step 9: Initialize access control
def initializeAccessControl():
    # Initialize the access control mechanism
    # Define permissions and restrictions for different parts of the graph
    or data
    # Return the initialized access control object
    return access_control
# Step 10: Check user interaction
def userInteractionNotComplete():
    # Check if the user's interaction is complete
    # Determine if there are more queries or if the user has finished
    interacting
    return not userFinishedInteraction # Example condition
# Step 11: Get user query
def getUserQuery():
    # Get the user's query or request for information
    # Return the user query
    return userQuery
# Step 12: Get indirect feedback
def getIndirectFeedback(userBehavior, queryPatterns, contextualInfo):
```

```
    # Gather indirect feedback from the user based on their behavior, query
    patterns, and contextual information
    # Analyze the user's actions, preferences, or feedback to refine their
    requirements
    # Return the indirect feedback information
    return indirectFeedback
# Step 13: Fine-tune query
def fineTuneQuery(userQuery, indirectFeedback):
    # Modify or refine the user's query based on the indirect feedback
    # Adjust the query to better match the user's preferences or requirements
    # Return the fine-tuned query
    return personalizedQuery
# Step 14: Graph traversal
def graphTraversal(graph, personalizedQuery):
    # Traverse the graph based on the personalized query
    # Explore the graph and retrieve relevant data or paths
   # Return the relevant data obtained from the graph traversal
   return relevantData
# Step 15: Filter data using access control
def filterData(access_control, relevant_data):
    # Apply access control to filter the relevant data based on the user's
    permissions and restrictions
    # Filter the data to ensure the user can only access authorized information
   filtered_data = access_control.filterAccess(relevant_data)
   return filtered_data
# Step 16: Display results
def displayResults(filtered_data):
   # Display the filtered data to the user as the final results
   # Present the relevant information to the user in a suitable format
   showResults(filtered_data)
# Step 17: Encrypt and authenticate data
def encryptAndAuthenticateData(data):
   # Apply encryption and authentication mechanisms to secure the data
   # Encrypt the data to protect its confidentiality
   encrypted_data = encryptData(data)
   # Authenticate the data to ensure its integrity and origin
    authenticated_data = authenticateData(encrypted_data)
    return authenticated_data
```

Pseudocode 2

```
def geneticAlgorithmOptimization(populationSize, terminationCondition,
initializePopulation, evaluateFitness, selection, crossover, mutate,
replacePopulation, getBestIndividual):
    # Initialization
    population = initializePopulation(populationSize)
    evaluateFitness(population)
    # Main loop
    while not terminationCondition():
        # Selection
        parents = selection(population)
        # Reproduction (Crossover and Mutation)
```

```
        offspring = crossover(parents)
        mutate(offspring)
        # Evaluation
        evaluateFitness(offspring)
        # Replacement
        population = replacePopulation(population, offspring)
        # Return best solution
    bestIndividual = getBestIndividual(population)
    return bestIndividual
# Example usage:
populationSize = N
def terminationCondition():
    # Define your termination condition here
    # Return True if the condition is met, False otherwise
    return ...
def initializePopulation(size):
    # Initialize and return a population of size 'size'
    return ...
def evaluateFitness(population):
    # Evaluate the fitness of each individual in the population
    # Update the fitness value of each individual accordingly
    return ...
def selection(population):
    # Perform selection to choose parents from the population
    # Return the selected parents
    return ...
def crossover(parents):
    # Perform crossover operation on the selected parents
    # Return the offspring produced by crossover
    return ...
def mutate(offspring):
    # Perform mutation operation on the offspring
    return ...
def replacePopulation(population, offspring):
    # Replace the population with the offspring
    # Return the new population
    return ...
def getBestIndividual(population):
    # Find and return the best individual from the population
    return ...
# Call the geneticAlgorithmOptimization function
bestSolution = geneticAlgorithmOptimization(
    populationSize,
    terminationCondition,
    initializePopulation,
    evaluateFitness,
    selection,
    crossover,
    mutate,
    replacePopulation,
    getBestIndividual
)
```

Pseudocode 3

```
function indirectFeedbackAlgorithm():
    # Initialization
    retrieveInitialData()
    while userInteractionNotComplete():
        userQuery = getUserQuery()
        # Step 1: Infer User Preferences
        inferredPreferences = inferUserPreferences(userBehavior,
        queryPatterns, contextualInfo)
        # Step 2: Refine User Query
        refinedQuery = refineQuery(userQuery, inferredPreferences)
        # Step 3: Retrieve Data
        retrievedData = retrieveData(refinedQuery)
        # Step 4: Display Results
        displayResults(retrievedData)
        # Step 5: Gather User Feedback
        userFeedback = gatherUserFeedback(retrievedData)
        # Step 6: Update Retrieval Process
        updateRetrievalProcess(retrievedData, userFeedback)
    # Finalize retrieval
    finalizeRetrieval()
# Step 1: Initialization - Retrieve Initial Data
function retrieveInitialData():
    # Perform necessary operations to retrieve and store initial data
    ...
# Step 2: Infer User Preferences
function inferUserPreferences(userBehavior, queryPatterns, contextualInfo):
    # Analyze user behavior, query patterns, and contextual information
    # to infer user preferences
    ...
    return inferredPreferences
# Step 3: Refine User Query
function refineQuery(userQuery, inferredPreferences):
    # Refine the user query based on inferred preferences
    ...
    return refinedQuery
# Step 4: Retrieve Data
function retrieveData(refinedQuery):
    # Perform the data retrieval process using the refined query
    ...
    return retrievedData
# Step 5: Display Results
function displayResults(retrievedData):
    # Present the retrieved data to the user in a suitable format
    ...
    return
# Step 6: Gather User Feedback
function gatherUserFeedback(retrievedData):
    # Allow the user to provide feedback on the displayed results
    # and collect the feedback for further processing
    ...
    return userFeedback
```

```
# Step 6: Update Retrieval Process
function updateRetrievalProcess(retrievedData, userFeedback):
    # Update the retrieval process based on user feedback
    # to improve future retrieval results
    ...
    return
# Finalize Retrieval
function finalizeRetrieval():
    # Perform any necessary cleanup or finalization steps
    # after the retrieval process is complete
    ...
    Return
```

Pseudocode 4

```
# Define a class for the Graph-based Data Retrieval
class GraphBasedDataRetrieval:
    graph = initializeGraph() # Initialize an empty graph
    function __init__():
        retrieveInitialData()
        constructGraph()
    function constructGraph():
        # Construct the graph representation of interconnected data entities
        for entity in initialData:
            graph.addNode(entity)
            relatedEntities = findRelatedEntities(entity)
            for relatedEntity in relatedEntities:
                graph.addEdge(entity, relatedEntity)
    function findRelatedEntities(entity):
        # Use similarity measures and clustering techniques to find related
        entities
        ...
    function optimizeRetrieval(query):
        relevantNodes = graph.traverse(query) # Use a graph traversal algorithm
        return relevantNodes
# Main function for the retrieval process
function graphBasedRetrieval():
    retrievalSystem = GraphBasedDataRetrieval()
    while userInteractionNotComplete():
        userQuery = getUserQuery()
        # Step 1: Infer User Preferences
        inferredPreferences = inferUserPreferences(userBehavior, queryPatterns,
        contextualInfo)
        # Step 2: Refine User Query
        refinedQuery = refineQuery(userQuery, inferredPreferences)
        # Step 3: Optimize Data Retrieval using Graph
        relevantNodes = retrievalSystem.optimizeRetrieval(refinedQuery)
        # Step 4: Retrieve and Display Results
        retrievedData = retrieveData(relevantNodes)
        displayResults(retrievedData)
        # Step 5: Gather User Feedback
        userFeedback = gatherUserFeedback(retrievedData)
        # Step 6: Update Graph and Retrieval Process
```

```
        retrievalSystem.updateGraph(userFeedback)
    # Finalize retrieval
    retrievalSystem.finalizeRetrieval()
# Step 1: Initialization - Retrieve Initial Data
function retrieveInitialData():
    # Perform necessary operations to retrieve and store initial data
    ...
# Step 2: Infer User Preferences
function inferUserPreferences(userBehavior, queryPatterns, contextualInfo):
    # Analyze user behavior, query patterns, and contextual information
    # to infer user preferences
    ...
    return inferredPreferences
# Step 3: Refine User Query
function refineQuery(userQuery, inferredPreferences):
    # Refine the user query based on inferred preferences
    ...
    return refinedQuery
# Step 4: Retrieve Data
function retrieveData(relevantNodes):
    # Perform the data retrieval process using the relevant nodes from the graph
    ...
    return retrievedData
# Step 5: Display Results
function displayResults(retrievedData):
    # Present the retrieved data to the user in a suitable format
    ...
    return
# Step 6: Gather User Feedback
function gatherUserFeedback(retrievedData):
    # Allow the user to provide feedback on the displayed results
    # and collect the feedback for further processing
    ...
    return userFeedback
# Step 6: Update Graph and Retrieval Process
function updateRetrievalProcess(userFeedback):
    # Update the graph and retrieval process based on user feedback
    ...
    return
# Finalize Retrieval
function finalizeRetrieval():
    # Perform any necessary cleanup or finalization steps
    ...
    return
# Start the graph-based retrieval process
graphBasedRetrieval()
```

Pseudocode 5

```
def secureAccessAlgorithm():
    # Initialization
    initializeAccessControl()
    while not userInteractionComplete():
```

```
        userQuery = getUserQuery()
        # Access Control
        authorizedData = applyAccessControl(userQuery)
        displayResults(authorizedData)
    # Algorithm complete
    print("Secure access algorithm complete.")
def initializeAccessControl():
    """
    Initializes the access control mechanism.
    Perform any necessary setup and configuration.
    """
    # TODO: Implement access control initialization logic here
    print("Access control initialized.")
def userInteractionComplete():
    """
    Checks if the user interaction is complete.
    Returns True if the interaction is complete, False otherwise.
    """
    # TODO: Implement logic to check if the user interaction is complete
    return False # Placeholder return value
def getUserQuery():
    """
    Retrieves a user query or input.
    Returns the user query as a string.
    """
    # TODO: Implement logic to retrieve user query
    userQuery = input("Enter your query: ")
    return userQuery
def applyAccessControl(userQuery):
    """
    Applies access control rules to the user query.
    Returns the authorized data based on the access control rules.
    """
    # TODO: Implement access control logic here
    authorizedData = None # Placeholder authorized data
    return authorizedData
def displayResults(authorizedData):
    """
    Displays the results of the authorized data.
    """
    # TODO: Implement logic to display the authorized data
    print("Authorized data: ", authorizedData)
# Execute the secure access algorithm
secureAccessAlgorithm()
```

Pseudocode 6

```
function secureDataManagementAlgorithm():
    initializeEncryption() # Initialize the encryption system
    initializeAuthentication() # Initialize the authentication system
    while userInteractionNotComplete():
        userQuery = getUserQuery() # Get user input/query
        authenticatedUser = authenticateUser(userQuery) # Authenticate the user
```

```
        if authenticatedUser is not None:
            encryptedData = encryptData(userQuery, authenticatedUser)
            # Encrypt the user query with authenticated user's credentials
            displayResults(encryptedData) # Display the encrypted results to
            the user
        else:
            displayAuthenticationError() # Display authentication error message
    displayAlgorithmCompleteMessage() # Display completion message once user
    interaction is complete
```

Pseudocode 7

```
def secureDataManagementAlgorithm():
    initializeBlockchain() # Initialize the blockchain network
    while userInteractionNotComplete():
        userQuery = getUserQuery()
        # Authentication
        authenticatedUser = authenticateUser(userQuery)
        if authenticatedUser is None:
            displayAuthenticationError()
            continue
        # Encryption
        encryptedData = encryptData(userQuery, authenticatedUser)
        # Store encrypted data on the blockchain
        transactionId = storeDataOnBlockchain(encryptedData)
        if transactionId is None:
            displayTransactionError()
            continue
        # Display the transaction ID to the user
        displayTransactionId(transactionId)
    # Algorithm complete
    displayAlgorithmCompleteMessage()
def initializeBlockchain():
    # Connect to the blockchain network
    # Set up necessary configurations and credentials
    # Initialize necessary data structures and smart contract instances
def getUserQuery():
    # Retrieve the user's query from the input interface
    # Validate and sanitize the input
def authenticateUser(userQuery):
    # Perform user authentication based on the query
    # Validate user credentials and permissions
    # Use appropriate authentication mechanisms such as username/password,
    API keys, or digital signatures
    # Return authenticated user object or None if authentication fails
def encryptData(userQuery, authenticatedUser):
    # Apply encryption techniques to protect the data
    # Utilize encryption algorithms (e.g., AES, RSA) and keys specific to the
    user
    # Ensure proper handling of key generation, key storage, and encryption/
    decryption processes
    # Return the encrypted data
def storeDataOnBlockchain(encryptedData):
```

```
    # Create a transaction to store the encrypted data on the blockchain
    # Utilize smart contracts and transaction mechanisms provided by the
    blockchain platform
    # Handle any necessary error conditions and return the transaction ID or
    None if storing fails
def displayTransactionId(transactionId):
    # Display the transaction ID to the user for future reference
def userInteractionNotComplete():
    # Check if the user interaction is complete
    # This can be based on a condition or user input
    # Return True if interaction is not complete, False otherwise
def displayAuthenticationError():
    # Display an authentication error message to the user
def displayTransactionError():
    # Display a transaction error message to the user
def displayAlgorithmCompleteMessage():
    # Display a message indicating the completion of the algorithm
def displayResults(encryptedData):
    # Display the encrypted data to the user
```

Appendix 2

Table 1

Scenario	Existing Method	Proposed Algorithm	Comparative Analysis
Scenario A	Predefined query-based retrieval	Evolutionary indirect feedback mechanism	Existing method accuracy: 75%. Proposed algorithm accuracy: 92%. The proposed algorithm outperforms the existing method with a 17% improvement in accuracy. It adapts to user feedback and achieves better retrieval results.
	K-nearest neighbors (KNN)		Existing method accuracy: 80%. The proposed algorithm surpasses KNN in accuracy by 12%. It learns from feedback, leading to superior results.
	Support vector machines (SVM)		Existing method accuracy: 85%. The proposed algorithm achieves a 7% increase in accuracy over SVM due to its graph-based retrieval and feedback capabilities.

(*Continued*)

Table 1 Continued

Scenario	Existing Method	Proposed Algorithm	Comparative Analysis
Scenario B	Role-based access control for linked data	Access control + graph-based retrieval	Existing method scalability: Limited. Proposed algorithm scalability: High. The proposed algorithm exhibits high scalability in handling big-linked spatiotemporal data compared to role-based access control.
	Access control lists (ACLs)		Existing method granularity: Coarse. Proposed algorithm granularity: Fine. The proposed algorithm offers finer access control granularity compared to ACLs.
	Attribute-based access control (ABAC)		Existing method retrieval efficiency: Moderate. Proposed algorithm retrieval efficiency: High. The proposed algorithm's graph-based retrieval achieves higher efficiency than ABAC.
Scenario C	Triple DES, RSA, Blowfish for Encryption	Encryption + fine-grained access control	Existing method security: Strong. Proposed algorithm security: enhanced. The proposed algorithm enhances security by incorporating fine-grained access control with encryption.
	Attribute-based encryption (ABE)		Existing method flexibility: Limited. Proposed algorithm flexibility: High. The proposed algorithm provides greater flexibility with attribute-based access control.
	Proxy re-encryption (PRE)		Existing method access control: Partial. Proposed algorithm access control: comprehensive. The proposed algorithm's access control includes user roles and permissions, unlike PRE.

(*Continued*)

Table 1 Continued

Scenario	Existing Method	Proposed Algorithm	Comparative Analysis
Scenario D	Traditional access control models	Indirect evolutionary feedback + graph-based techniques	Existing method adaptability: Low. Proposed algorithm adaptability: High. The proposed algorithm adapts dynamically to evolving user preferences, providing higher adaptability than traditional models.
	Role-based access control with temporal constraints		Existing method personalization: Moderate. Proposed algorithm personalization: High. The proposed algorithm's indirect feedback leads to personalized data retrieval compared to temporal role-based access control.
	Attribute-based access control with context awareness		Existing method relationships capture: Limited. Proposed algorithm relationships capture: Enhanced. The proposed algorithm captures complex data relationships better than context-aware attribute-based access control.
Scenario E	Role-based access control in distributed systems	Access control + graph-based retrieval	Existing method query handling: Inefficient. Proposed algorithm query handling: Efficient. The proposed algorithm's graph-based retrieval efficiently handles complex queries compared to role-based access control.
	Capability-based access control		Existing method security: Strong. Proposed algorithm security: Comparable. The proposed algorithm maintains comparable security while offering enhanced retrieval capabilities.
	Distributed hash tables (DHTs)		Existing method data sharing: Yes. Proposed algorithm data sharing: Yes. Both methods support data sharing, but the proposed algorithm provides access control as well.

Table 2

Scenario	Existing Method	Proposed Algorithm	Comparative Analysis
Scenario A	Predefined query-based retrieval	Evolutionary indirect feedback mechanism	Predefined query-based retrieval lacks adaptability, while the proposed algorithm learns from feedback, offering enhanced accuracy and flexibility in spatiotemporal data retrieval.
	K-nearest neighbors (KNN)		KNN is a traditional method for retrieval, but it does not adapt well to evolving data and lacks the benefits of evolutionary feedback provided by the proposed algorithm.
	Support vector machines (SVM)		SVM is another commonly used retrieval method, but it struggles to handle complex spatiotemporal data relationships, unlike the proposed graph-based approach.
Scenario B	Role-based access control for linked data	Access control + graph-based retrieval	Role-based access control lacks efficient retrieval and scalability, while the proposed algorithm combines access control with graph-based retrieval, improving accuracy and scalability in big-linked spatiotemporal data handling.
	Access control lists (ACLs)		ACLs are commonly used for access control but do not address the retrieval challenges in big-linked spatiotemporal data that the proposed algorithm handles effectively.
	Attribute-based access control (ABAC)		ABAC improves access control granularity but does not offer efficient graph-based retrieval capabilities present in the proposed algorithm.

(*Continued*)

Table 2 Continued

Scenario	Existing Method	Proposed Algorithm	Comparative Analysis
Scenario C	Triple DES for encryption	Encryption + fine-grained access control	Triple DES provide encryption but lack fine-grained access control. In contrast, the proposed algorithm integrates these encryption methods with access control, enhancing overall security and providing comprehensive solutions for secure data retrieval and sharing.
	Attribute-based encryption (ABE)		ABE enhances encryption with attribute-based access control, but it not offer the same level of fine-grained control as the proposed algorithm.
	Proxy re-encryption (PRE)		PRE allows intermediaries to re-encrypt data, but it not covers the comprehensive access control capabilities of the proposed algorithm, which include user roles and permissions.
Scenario D	Traditional access control models	Indirect evolutionary feedback + graph-based techniques	Traditional access control models are not suitable for dynamic spatiotemporal data, whereas the proposed algorithm incorporates indirect evolutionary feedback and graph-based techniques, enabling more accurate and personalized data retrieval catering to evolving user needs.
	Role-based access control with temporal constraints		Role-based access control with temporal constraints provides some adaptability, but it does not capture evolving user preferences and relationships as effectively as the proposed algorithm.

(*Continued*)

Table 2 Continued

	Attribute-based access control with context awareness		Attribute-based access control with context awareness provide more flexibility but could still lack the dynamic feedback mechanism of the proposed algorithm.
Scenario E	Role-based access control in distributed systems	Access control + graph-based retrieval	Role-based access control in distributed systems lacks efficient retrieval, whereas the proposed algorithm combines access control with graph-based retrieval, efficiently handling complex queries and retrieving relevant spatiotemporal data.
	Capability-based access control		Capability-based access control improves security but not offer the same level of retrieval capabilities as the proposed graph-based approach.
	Distributed hash tables (DHTs)		DHTs facilitate data sharing but not address the access control and retrieval challenges in distributed big-linked spatiotemporal data that the proposed algorithm does.

References

[1] A. Bifet, G. de Francisci Morales, J. Read, G. Holmes, and B. Pfahringer, "Efficient Online Evaluation of Big Data Stream Classifiers," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, in KDD '15. New York, NY, USA: Association for Computing Machinery, Dec. 2015, pp. 59–68. doi: 10.1145/2783258.2783372.

[2] M. G., T. C., L. D., and S. R, *Big Data Security Intelligence for Healthcare Industry 4.0*. in Springer Series in Advanced Manufacturing. Cham: Springer, 2017.

[3] X. Shi, R. Qiu, Z. Ling, F. Yang, H. Yang, and X. He, "Spatio-Temporal Correlation Analysis of Online Monitoring Data for Anomaly Detection and Location in Distribution Networks," *IEEE*

Trans. Smart Grid, vol. 11, no. 2, pp. 995–1006, Mar. 2020, doi: 10.1109/TSG.2019.2929219.

[4] A. Bates and W. U. Hassan, "Can Data Provenance Put an End to the Data Breach?," *IEEE Secur. Priv.*, vol. 17, no. 4, pp. 88–93, Jul. 2019, doi: 10.1109/MSEC.2019.2913693.

[5] S. Chun, S. J. Oh, R. Sampaio de Rezende, Y. Kalantidis, and D. Larlus, "Probabilistic Embeddings for Cross-Modal Retrieval," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2021, pp. 8411–8420. doi: 10.1109/CVPR46437.2021.00831.

[6] Z. Li, W. Xu, H. Shi, Y. Zhang, and Y. Yan, "Security and Privacy Risk Assessment of Energy Big Data in Cloud Environment," *Comput. Intell. Neurosci.*, vol. 2021, p. e2398460, Oct. 2021, doi: 10.1155/2021/2398460.

[7] V. Alieksieiev and B. Andrii, "Information Analysis and Knowledge Gain within Graph Data Model," in *2019 IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT)*, Sep. 2019, pp. 268–271. doi: 10.1109/STC-CSIT.2019.8929812.

[8] S. Behera and J. R. Prathuri, "Application of Homomorphic Encryption in Machine Learning," in *2020 2nd PhD Colloquium on Ethically Driven Innovation and Technology for Society (PhD EDITS)*, Aug. 2020, pp. 1–2. doi: 10.1109/PhDEDITS51180.2020.9315305.

[9] A. C. F. Chan and C. Castelluccia, "A security framework for privacy-preserving data aggregation in wireless sensor networks," *ACM Trans. Sens. Netw. TOSN*, vol. 7, no. 4, 2011, doi: 10.1145/1921621.1921623.

[10] L. Avigad and O. Goldreich, "Testing Graph Blow-Up," in *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation: In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, O. Goldreich, Ed., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2011, pp. 156–172. doi: 10.1007/978-3-642-22670-0_18.

[11] M. Bellare et al., "On Probabilistic versus Deterministic Provers in the Definition of Proofs of Knowledge," in *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation: In Collaboration with Lidor*, O. G. David Zuckerman, Ed., Berlin, Heidelberg: Springer, 2011, pp. 114–123. doi: 10.1007/978-3-642-22670-0_14.

[12] S. Blyumin, A. Pogodaev, and E. Khabibullina, "Graph-structural Modeling of Some Special Organizational Systems," in *2020 2nd International Conference on Control Systems, Mathematical Modeling, Automation and Energy Efficiency (SUMMA)*, Aug. 2020, pp. 279–283. doi: 10.1109/SUMMA50634.2020.9280724.

[13] S. S. Alotaibi, "Registration Center Based User Authentication Scheme for Smart E-Governance Applications in Smart Cities," *IEEE Access*, vol. 7, pp. 5819–5833, 2019, doi: 10.1109/ACCESS.2018.2884541.

[14] G. Avalle, F. De Pace, C. Fornaro, F. Manuri, and A. Sanna, "An Augmented Reality System to Support Fault Visualization in Industrial Robotic Tasks," *IEEE Access*, vol. 7, pp. 132343–132359, 2019, doi: 10.1109/ACCESS.2019.2940887.

[15] B. Bordel, R. Alcarria, and T. Robles, "Lightweight encryption for short-range wireless biometric authentication systems in Industry 4.0," *Integr. Comput.-Aided Eng.*, vol. Preprint, no. Preprint, pp. 1–21, Jan. 2021, doi: 10.3233/ICA-210673.

[16] V. R. Catherine and A. S. Nargunam, "Multi authority Ciphertext-Policy Attribute-based encryption for security enhancement in the cloud storage unit," *Sustain. Energy Technol. Assess.*, vol. 53, p. 102556, 2022.

[17] A. Bakdi, A. Hentout, and H. Boutami, "Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control[J," *Robot. Auton. Syst.*, vol. 89, pp. 95–109, 2017.

[18] S. Olyaee, R. Ebrahimpur, and S. Esfandeh, "A hybrid genetic algorithm-neural network for modeling of periodic nonlinearity in three-longitudinal-mode laser heterodyne interferometer," in *2013 21st Iranian Conference on Electrical Engineering (ICEE)*, May 2013, pp. 1–5. doi: 10.1109/IranianCEE.2013.6599790.

[19] L. Cheng, H. Z. G, and Y. Lin, "Recurrent Neural Network for Non-Smooth Convex Optimization Problems With Application to the Identification of Genetic Regulatory Networks[J," *IEEE Trans. Neural Netw.*, vol. 22, no. 5, pp. 714–26, 2011.

[20] H. J. X, M. M. Y, and W. K, "Product modeling design based on genetic algorithm and BP neural network[J]," *Neural Comput. Appl.*, vol. 33, no. 9, pp. 4111–4117, 2021.

[21] S. Ding, S. Qu, and Y. Xi, "A long video caption generation algorithm for big video data retrieval[J]," *Future Gener. Comput. Syst.*, vol. 93, pp. 583–595, 2019.

[22] C. C. Aggarwal, P. S. Yu, J. Han, and J. Wang, "- A Framework for Clustering Evolving Data Streams," in *Proceedings 2003 VLDB Conference*,

J.-C. Freytag, P. Lockemann, S. Abiteboul, M. Carey, P. Selinger, and A. Heuer, Eds., San Francisco: Morgan Kaufmann, 2003, pp. 81–92. doi: 10.1016/B978-012722442-8/50016-1.

[23] A. E. Barinov and A. A. Zakharov, "Clustering using a random walk on graph for head pose estimation," in *2015 International Conference on Mechanical Engineering, Automation and Control Systems (MEACS)*, Sep. 2015, pp. 1–5. doi: 10.1109/MEACS.2015.7414876.

[24] R. C. de Amorim, "Constrained clustering with Minkowski Weighted K-Means," in *2012 IEEE 13th International Symposium on Computational Intelligence and Informatics (CINTI)*, Aug. 2012, pp. 13–17. doi: 10.1109/CINTI.2012.6496753.

[25] S. Alhasan, G. Abdul-Salaam, L. Bayor, and K. Oliver, "Intrusion Detection System Based on Artificial Immune System: A Review," in *2021 International Conference on Cyber Security and Internet of Things (ICSIoT)*, Sep. 2021, pp. 7–14. doi: 10.1109/ICSIoT55070.2021.00011.

[26] A. Bala, I. Ismail, R. Ibrahim, and S. M. Sait, "Applications of Metaheuristics in Reservoir Computing Techniques: A Review," *IEEE Access*, vol. 6, pp. 58012–58029, 2018, doi: 10.1109/ACCESS.2018.2873770.

[27] N. Aljohani, J. Shelton, and K. Roy, "Authentication Based on Touch Patterns Using an Artificial Immune System," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, Sep. 2021, pp. 1–8. doi: 10.1109/SSCI50451.2021.9660096.

[28] Y. Cho et al., "A Secure Three-Factor Authentication Protocol for E-Governance System Based on Multiserver Environments," *IEEE Access*, vol. 10, pp. 74351–74365, 2022, doi: 10.1109/ACCESS.2022.3191419.

[29] M. Borassi, A. Epasto, S. Lattanzi, S. Vassilvitskii, and M. Zadimoghaddam, "Sliding Window Algorithms for k-Clustering Problems." arXiv, Oct. 23, 2020. doi: 10.48550/arXiv.2006.05850.

[30] M. Abdullah and M. Hadzikadic, "Sentiment Analysis of Twitter Data: Emotions Revealed Regarding Donald Trump during the 2015-16 Primary Debates," in *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, Aug. 2017, pp. 760–764. doi: 10.1109/ICTAI.2017.00120.

[31] R. Chauhan and S. S. Heydari, "Polymorphic Adversarial DDoS attack on IDS using GAN," in *2020 International Symposium on Networks, Computers and Communications (ISNCC*, Jul. 2020, pp. 1–6. doi: 10.1109/ISNCC49221.2020.9297264.

[32] Y. Hui and L. Zesong, "Research on Real-time Analysis and Hybrid Encryption of Big Data," in *2019 2nd International Conference on Artificial Intelligence and Big Data (ICAIBD)*, Feb. 2019, pp. 52–55. doi: 10.1109/ICAIBD.2019.8836992.
[33] S. Ahmed, M. M. Alshater, A. E. Ammari, and H. Hammami, "Artificial intelligence and machine learning in finance: A bibliometric review," *Res. Int. Bus. Finance*, vol. 61, p. 101646, Oct. 2022.
[34] D. Atienza, P. Larranaga, and C. Bielza, "Hybrid semiparametric Bayesian networks," *TEST*, vol. 31, no. 2, pp. 299–327, 2022.

Biographies

Song Li obtained a Master's degree in computer application technology from Nanjing University of Aeronautics and Astronautics in China. At present, he is pursuing a Ph.D. degree in cyberspace security at Guilin University of Electronic Technology, also in China. His primary research interests revolve around data security and blockchain technology.

WenFen Liu earned a Ph.D. in cryptography from the Information Engineering University of the People's Liberation Army in 1999. In 2017,

she joined Guilin University of Electronic Technology as a distinguished faculty member. Currently, she holds the position of Doctoral Supervisor and serves as a Professor at the School of Computer and Information Security, Guilin University of Electronic Technology. Her research focuses primarily on privacy protection, statistical analysis technology for big data security, and the design and analysis of cryptographic algorithms.

Yan Wu obtained a Bachelor's degree in Management from Guilin University of Electronic Technology in 2006, and now she works in the Unit 95795 of People's Liberation Army. Her research interests mainly include big data and multimedia applications.

Jie Zhao received a B.Eng. in the School of Information and Management Science at Henan Agricultural University, Zhengzhou, China. He is currently studying for a M.Eng. in electronic science and Technology at the School of Computer and Information Security, Guilin University of Electronic Technology, China. His research interests focus on blockchain.