Journal of Engineering and Technology for Industrial Applications

ITEGAM-JETIA

Manaus, v.12 n.58, p. 610-620. March/April, 2026.

DOI: https://doi.org/10.5935/jetia.v12i58.3203

ISSN ONLINE: 2447-0228

RESEARCH ARTICLE OPEN ACCESS

EFFICIENT EXTRACTION OF PATTERNS AND INSIGHTS FROM LARGE-SCALE DATA FROM DISTRIBUTED SYSTEM USING DEEP LEARNING AND LSTM

K. Syed Kousar Niasi*[1], K. Saravanakumar [2], Anand Viswanathan [3] and K. Ravikumar[4]

[1]Assistant Professor, Department of Computer Science, Jamal Mohamed College (Affiliated to the Bharathidasan University), Tiruchirappalli, India.
[2]Associate Professor, Department of Computer Science and Engineering, Sri Eshwar College of Engineering, Coimbatore, India.
[3]Professor, Department of Information Technology, Ponjesly College of Engineering, Nagercoil, India.
[4]Professor, Department of Information Technology, Dhanalakshmi Srinivasan College of Engineering and Technology, Chennai, India.

[1]https://orcid.org/0009-0001-9993-6196, [2]https://orcid.org/0009-0003-6353-7702
[3]https://orcid.org/0000-0002-7754-5346, [4]https://orcid.org/0000-0003-2826-289X

Email: *syedkousarniasi@gmail.com, saravanakumar.phdauc@gmail.com, itsanandmtech@gmail.com, ravikumarcsephd@gmail.com

ARTICLE INFO

Article History
Received: December 30, 2025
Reviewed: January 1, 2026
Accepted: January 19, 2026
Published: March 31, 2026

Keywords:
Data analysis
Data mining
Deep learning
Distributed systems
Parallel computing

ABSTRACT

Extraction of useful insights and patterns from massive datasets is impossible without data mining. The widespread use of Internet services and the subsequent creation of vast quantities of data have ushered in the modern era of big data. Every aspect of human existence generates copious amounts of data. Data analysis and utilization processes need to take into consideration a growing number of factors to deal with the growing volume and complexity of today's data sets. The importance of deep learning in data mining rises in proportion to the problem's complexity. Traditional data mining methods face major hurdles due to the ever-increasing number and complexity of data. To address this issue, we offer a unique distributed data mining strategy that utilizes deep learning and Long Short-Term Memory (LSTM) systems. In this study, we combine deep learning with distributed computing to provide a powerful tool for data mining. In order to capture long-term relationships in sequential data, the proposed model uses recurrent neural networks (RNNs) of the LSTM kind. Since LSTM networks are so effective with time-series and sequential data, they may be used in a wide variability of data mining tasks. We create a parallel computing framework that pools the processing power of a cluster's many nodes to facilitate decentralized operations. The training and inference of the LSTM-based data mining model are sped up by the distributed design, which also facilitates the efficient processing of big datasets. We compare the proposed model to standard data mining techniques and show that it outperforms them on a number of real-world datasets. The outcomes demonstrate the superior accuracy of 99.5% and efficiency of our deep learning-based method for identifying useful patterns and making predictions. The findings validate the model's horizontal scalability and the benefits of distributed computing, guaranteeing its practical application in large data settings.

I. INTRODUCTION

In this era of big data, we cannot function without the capability to investigate and interpret that data. The usage of big data is on the rise, and it is already being put to work in a variety of fields, from sickness and reward prediction to intelligent transportation system assistance to decision-making aid for businesses to the study of consumer psychology. People's expectations of data outputs are rising, prompting the development of more robust approaches like categorization, clustering, and neural networks. On the other hand, there is both helpful and worthless information in a large data set. Extracting valuable insights from massive datasets while discarding irrelevant information is a critical challenge in modern data mining.

To keep up with the growing need for data analysis in today's interconnected world, scientists and engineers have been investigating the potential of deep learning in data processing. A time series is a collection of data points that has been sequentially organized by time. These sites are often chosen at regular intervals. Time-series data categorization is a major data mining profession is facing a significant difficulty [1] The distributed data mining method [2] has to be geared toward the centrally acquired data and must perform analysis on the data in a decentralized fashion. Let's assume that the dataset has a large dimensionality and that the data mining approach requires a lot of processing power and time. Then, utilizing a distributed network of computers, the problem is solved using methods of parallel knowledge analysis [3], [4]. So, we use machine learning methods [5], [6] to look at the decentralized dataset and see what we can learn from it. Clustering is an efficient method that analyzes data similarities to find unexpected outcomes. Similar pieces of information are grouped together by the clustering procedure [7], [8] Attributed information is data that has been partitioned in a particular way, such as density, centroid, hierarchical, or spectral.

However, there may be significant complication in data privacy and security problems due to the clustering. Vertical and horizontal dimensions, also known as heterogeneous and homogeneous viewpoints, are used in DDM's analysis of the data [9]. The current clustering technique uses assumptions of data similarity to address problems with dispersed clustering. The scattered data in a challenging circumstance is analyzed afterwards by looking into heterogeneous data.It takes more work to reduce the computing complexity while keeping the important data while working with a heterogeneous dataset due to its big dimension [10]. During this phase, the system integrates information from several sources to facilitate efficient discovery of new information. Sensitive information is among the data sets that have been dispersed. Therefore, the fusing process must take into account concerns about data privacy and security.

Therefore, it is necessary to construct a powerful distributed data mining system to deal with the massive amount of data with minimal processing complexity, such as problems with data security and dimensionality. To get around this problem and reduce the dimensionality of the data, many academics turn to techniques like correlation and principal component analysis. Due to the non-linear interacting problems inherent in this procedure, only a small subset of relevant data is taken into account during clustering. Existing techniques of grouping comparable data, however, have substantial mistake rates. Problems are put to rest when the mayfly-optimized deep learning algorithm is used to study the data's features and cope with the data redundancy issue. Furthermore, the neural network utilizes the knowledge from the local dataset to discover universal patterns for targeted issues. Previous learning and feature connection are mined for the global patterns. The network then makes use of the optimization method to tweak the network's settings, reducing the severity of the optimization problem and the number of false positives.

II.RELATED WORKS

Machine learning techniques have been extensively used in diverse distributed computing thanks to the availability of organized information. Distributed computing may be sped up with the use of knowledge-based machine learning for applications that run in real time. To address the critical bottlenecks that plague real-time distributed systems, researchers have developed a Cognitive Distributed Learning System [11]. In order to compute belief nets, the hidden layer network is analyzed using learning techniques and certain related priors. Quick greedy and the wake-sleep strategy can be used for deep learning by taking use of associative memory. For discriminative outcomes in picture number classification, a model based on a three-layer architecture has been developed [12]. When discussing DCRN (Distributed Cognitive Radio Network) and DCS (Dynamic Channel Selection), reinforcement learning is an additional essential machine learning method. Using both a single agent and a group of agents, this machine learning technique takes use of the density of the network to achieve maximum fairness [13]. This kind of learning allows scientists to improve their performance in several areas without sacrificing productivity or quality in their machines. A self-motivated mechanized company may have a bright future based on capability, change, and self-organisation, as suggested by a hierarchical mechanical management structure [14].

The Case Based Reasoning (CBR) method is also used in conjunction with deep learning to categorize clinical data. Visual reasoning and an interface are used to display patient diagnosis data, allowing for both qualitative and quantitative analysis to be carried out. Multidimensional medical data, such as that pertaining to breast cancer, may be visually described by means of CBR [15]. In the field of cancer prediction, neural networks have become the standard tool. Examining the study reveals the cancer survival risk and engineering forecasts. The results of these trials show how effective neural network learning can be [16]. When used to health data, artificial neural networks (ANNs) can improve data delivery while cutting costs. In describing health-related real-world applications on diverse large medical datasets, ANN provides a conclusive analysis. Two examples of general-purpose ANN that produce reliable outcomes when applied to healthcare data are feed-forward and hybrid networks. The use of ANN for making decisions on patients' medical records has been used by both external and internal networks [17]. Artificial intelligence's (AI's) rising use in clinical data is changing the face of health data liberation in profound ways. The medical and scientific communities, among others, make extensive use of AI. Previously untapped opportunities for medical organizations have been identified thanks to AI strategies for producing equipment to wisdom and grasp information.

In [18], K-means clustering is used to keep data private in a decentralized system. The goal of this system is to control personal information in a decentralized setting. To deal with the local differential privacy issues introduced by highly altered data, the K-means clustering method is used. Then, with the help of efficient clustering, each individual user's local privacy is bolstered. In [19], the privacy protected K-means clustering method is used for managing data privacy in an unsecure setting. The external adversaries can be anticipated by the distributed clustering method thanks to the usage of elliptic curve encryption techniques. The ability to accurately forecast attacks aids in maintaining security and privacy with minimal computational complexity and overhead. To improve safety and confidentiality in smart electrical systems, [20] presents a privacy data availability clustering method. The data clustering technique is used in this paper in an effort to mitigate privacy leakage and distorted data concerns. To determine how far apart the data and the centroid computation are, the technique use the K-means algorithm. User behavior and privacy characteristics in social networks were predicted using a blockchain-based privacy algorithm in a distributed database (DEPLSEST) in [21]. Privacy in locally stored databases may be managed using synchronization procedures included in the blockchain with clustering algorithm. The consensus procedure is then used to update all relevant ledgers.

When compared to competing secure clustering algorithms, the protocol provides superior byzantine fault tolerance. The multi-core DBSCAN method for handling confidential network traffic information is presented in [22]. This method protects sensitive information without requiring special skills or knowledge from the opponents. The transmitted data is secure due to the inclusion of random noise. According to the research shown above, there are a number of obstacles to reaching high accuracy and lowering the error rate using current approaches. However, when dealing with heterogeneous data in a distributed setting, such solutions need a lot of computational resources and offer nothing in the way of security. As a result, the current infrastructure can't cope with the multi-resource data that comes with high-dimensional dimensions. The large dimensionality reduces the effectiveness of the clustering process as a whole and raises privacy concerns. As a result, the clustering and user authentication processes experience a discrepancy between the real and calculated numbers. This paper provides a solution to this problem by using Distributed Clustering with Optimised Deep Learning (DCODL) techniques. The effectiveness of the new DCODL approach is measured against previous studies.

III. METHODS AND MATERIALS

III.1 PROBLEM FORMULATION

Given a large-scale dataset consisting of sequential data, represented as a set of input sequences $X = \{x_1, x_2, \ldots, x_N\}$, where each $x_i = (x_{i1}, x_{i2}, \ldots, x_{im})$ represents a sequence of m data points, we aim to develop an efficient extraction model that utilizes deep learning and LSTM networks to uncover valuable patterns and insights from the data. Let $Y = \{y_1, y_2, \ldots, y_N\}$ denote the corresponding target sequences, where $Y = \{y_{i1}, y_{i2}, \ldots, yi_N$ represents the corresponding target sequence associated with x_i. The target sequences could be binary labels, regression values, or any other form of target variable. We propose a distributed system architecture that harnesses the computational resources of multiple nodes in a cluster to facilitate efficient processing of the large-scale dataset. Each node in the cluster collaboratively performs computations in parallel to accelerate the training and inference stages of the deep learning model. The deep learning model leverages LSTM networks, which are a type of recurrent neural network able to detect temporal relationships in sequential data. Let $f(x;\ \theta)$ represent the LSTM-based model, parameterized by θ, that maps an input sequence x to the corresponding predicted output sequence.

The model is trained using a suitable optimization algorithm, such as stochastic gradient descent, to minimize a given loss function $L(Y, f(X;\ \theta))$ that measures the discrepancy between the predicted output sequences and the true target sequences. To enable efficient extraction of patterns and insights, we define performance metrics such as accuracy, precision, recall, or any other suitable evaluation measures, depending on the specific data mining task. The objective is to maximize these performance metrics while ensuring efficient computation and resource utilization within the distributed system. By employing the proposed deep learning-based distributed data mining approach, we aim to demonstrate its superiority over traditional data mining methods in terms of accuracy, efficiency, and scalability. Through experimental evaluations on large-scale real-world datasets, Our proposed model is compared to industry standards, validating its ability to efficiently extract patterns and insights from the large-scale data while achieving high accuracy and scalability in a distributed computing environment.

III.2 DATA MINING

Data mining is a systematic process that involves several stages to extract valuable insights and patterns from large datasets as exposed in figure 1. The first step is identifying the data source, which could be a database, a data warehouse, or even unstructured data like text documents or web pages. Once the data source is determined, the next stage is data preprocessing, where the raw data is cleaned, integrated, and transformed to ensure its quality and consistency. After preprocessing, the target dataset is defined, specifying the subset of data that will be used for the mining process. This dataset is carefully selected based on the specific goals and objectives of the data mining project. Once the target dataset is established, an appropriate mining algorithm is chosen. According to the data and the outputs needed, these algorithms might range from decision trees and neural networks to clustering and association rule mining.

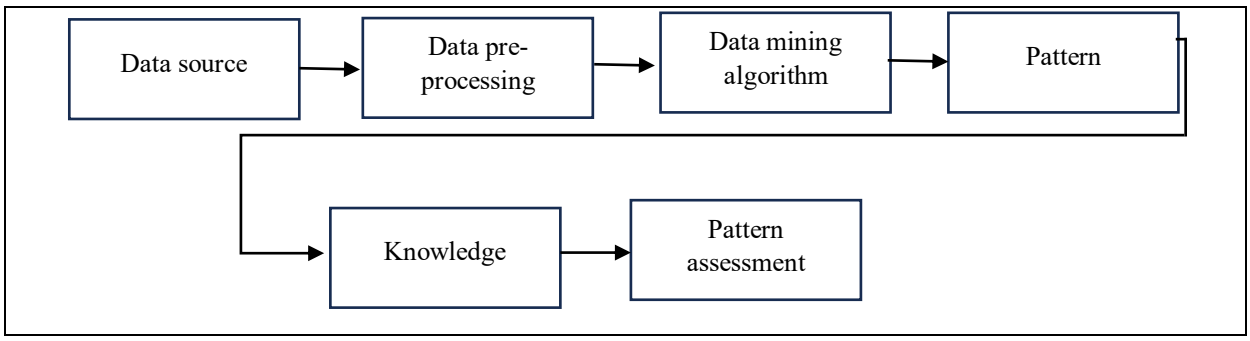

Figure 1: Basic data mining process.
Source: Authors, (2026).

The mining algorithm then analyses the target dataset, searching for meaningful patterns and relationships within the data. These patterns could be associations between items, sequential patterns, clusters of similar data points, or predictive models. Once the patterns are discovered, the next step is pattern assessment, where the quality and significance of the patterns are evaluated. This assessment may involve statistical measures, validation techniques, and comparison to domain knowledge or existing models. Finally, the last stage of the data mining process is to transform the discovered patterns into actionable knowledge. This knowledge can be used for decision making, prediction, or to gain a deeper understanding of the underlying data. It is important to note that data mining is an iterative process, and the knowledge gained from one cycle can be used to refine the process in subsequent iterations, leading to continuous improvement and enhanced insights.

III.3 METHODOLOGY

The proposed workflow begins with the collection of massive datasets from diverse sources, taking into account their volume and complexity. These datasets are then subjected to preprocessing, where cleaning and data quality assurance steps are performed to handle missing values, outliers, and ensure data integrity. Next, the large-scale dataset is partitioned into smaller subsets, which are distributed across multiple nodes within a distributed computing cluster. This enables parallel processing and efficient utilization of computational resources. For the distributed deep learning training phase, an LSTM-based model is initialized with appropriate hyperparameters. Each node independently trains the model using its allocated subset of data, while the model parameters are synchronized and updated iteratively across nodes to maintain consistency. This training process is repetitive until convergence or a predefined stopping criterion is met. In the distributed inference stage, the trained LSTM model is applied to perform inference on new data points. Inference tasks are distributed across nodes for parallel processing, and the results are collected and merged to obtain the final predictions.

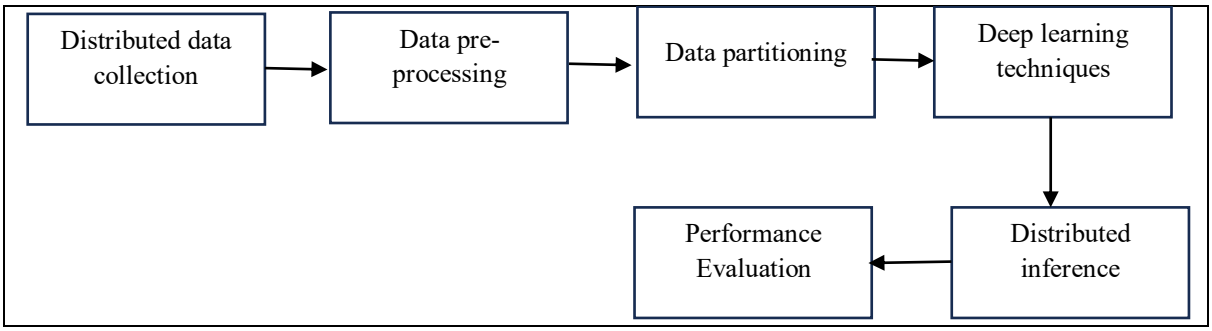

Figure 2: The overall structure of the proposed model.
Source: Authors, (2026).

The recital of the proposed deep learning-based distributed model as shown in the figure 2 is then evaluated and compared with traditional data mining techniques. Accuracy, efficiency, and other relevant metrics are assessed using real-world datasets to showcase the superiority of the deep learning approach in identifying valuable patterns and making predictions. The scalability and practical application of the distributed model are validated by examining its horizontal scalability as the dataset size and computational resources increase. The benefits of distributed computing, such as efficient processing of big datasets, are highlighted. Ultimately, the proposed approach's practical applicability in large data settings is emphasized, demonstrating its potential for real-world applications.

III.3.1 Data collection

Incident reports and traffic accidents, both examples of event data, are made up of isolated incidents that occurred at certain times and places. Generalizations about an event can be made about its nature, its place of occurrence, and its time of occurrence. A traffic accident, for instance, can be represented as the tuple (e, l, t), where e is the accident type, l is the location, and t is the time at which the accident takes place.

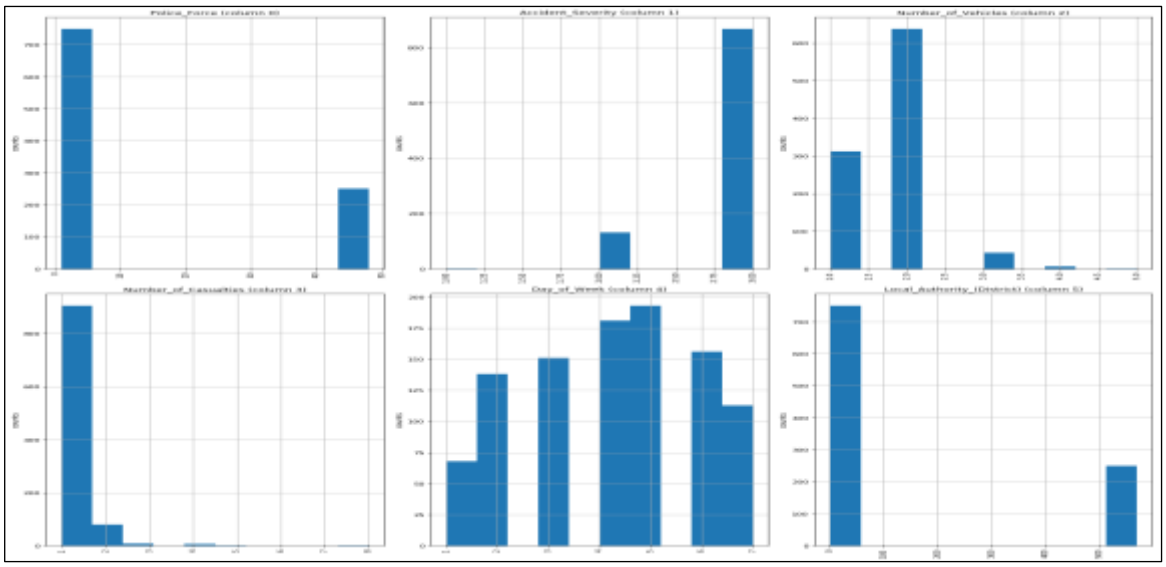

Figure 3: Distributed data collected from various resources.
Source: Authors, (2026).

The event data is depicted graphically in Fig. 3. There are three distinct events represented by the three distinct forms of the symbol. Many fields rely on event data, including those dealing with criminology (such as data on urban offences), epidemiology (such as data on disease outbreak events), transportation (such as data on traffic crashes), and social networks (such as data on conversations and trending themes).

III.3.2 Deep learning model

Long Short-Term Memory (LSTM) well-suited for tasks such as time series analysis, natural language processing, and speech recognition. LSTM models are particularly effective in handling sequential data, such as time series, text, or sensor data. In a distributed environment, where datasets may be massive and continuously streaming, LSTM models can capture long-term dependencies and relationships within the data, making them suitable for various data mining tasks. By combining LSTM models with distributed computing, the processing power of multiple nodes in a cluster can be harnessed. This allows for parallel processing of data, speeding up training and inference tasks. The distributed design helps in efficiently handling large-scale datasets, making it feasible to mine valuable insights from massive amounts of data. In a distributed environment, the dataset can be partitioned and distributed across nodes. Each node can perform local training on its allocated data subset using the LSTM model. This parallel training enables efficient utilization of computational resources and accelerates the training process, leading to faster convergence and improved scalability.

During training, the parameters of the LSTM model need to be synchronized and updated across distributed nodes to maintain consistency. Communication and synchronization protocols can be employed to exchange information and propagate parameter updates between nodes. This ensures that the model learns from the collective knowledge of the distributed dataset. Once the LSTM model is trained, it can be used for distributed inference on new data points. The inference tasks can be distributed across nodes for parallel processing, allowing for efficient prediction or pattern recognition across the distributed dataset. Distributed LSTM models exhibit scalability, allowing them to handle large-scale datasets and accommodate an increasing number of computational resources. Additionally, fault tolerance mechanisms can be incorporated to handle failures or node disruptions, ensuring the robustness and uninterrupted operation of the distributed data mining system. The figure 4 shows the structure of Long Short-Term Memory (LSTM).

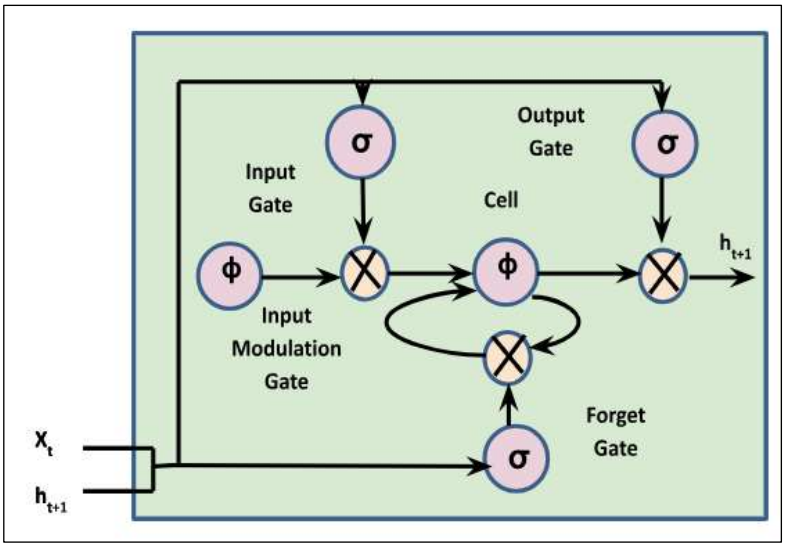

Figure 4: Structure of Long Short-Term Memory (LSTM).
Source: Authors, (2026).

The LSTM model consists of multiple memory cells, each capable of storing and updating information over time. It employs a set of gating mechanisms can manage the flow of data, enabling it to choose retain or discard data from earlier time stages. Let's denote the inputs at time step t as $x(t)$ and the hidden state (or memory) at time step t as $h(t)$. The LSTM model can be mathematically formulated as follows,

Input Gate ($i(t)$):

$$i(t) = \sigma(W_i x(t) + U_i h(t-1) + b_i) \quad (1)$$

Forget Gate ($f(t)$):

$$i(t) = \sigma(W_f x(t) + U_f h(t-1) + b_f) \quad (1)$$

Candidate Memory Cell ($\hat{C}(t)$):

$$\hat{C}(t) = tanh(W_c x(t) + U_c h(t-1) + b_c) \quad (2)$$

Memory Cell (c(t)):

$$c(t) = f(t) * c(t-1) + i(t) * \hat{C}(t) \quad (3)$$

Output Gate (o(t)):

$$o(t) = \sigma(W_o x(t) + U_o h(t-1) + b_O) \quad (4)$$

Hidden State (h(t)):

$$h(t) = o(t) * tanh(c(t)) \quad (5)$$

In the above equations, W_i, U_I, b_i, W_f, U_f, b_f, W_c, U_c, b_c, W_O, U_O, b_o are the weight matrices and bias vectors that need to be learned during the training process. σ denotes the sigmoid activation function, which squashes the values between 0 and 1, controlling the flow of information through the gates. The input gate ($i(t)$) determines how much of the candidate memory cell ($\hat{C}(t)$) should be added to the memory cell ($c(t)$), while the forget gate ($f(t)$) controls how much of the previous memory cell ($c(t-1)$) should be retained. The output gate ($o(t)$) determines how much of the memory cell ($c(t)$) should be passed to the hidden state ($h(t)$).

By utilizing the LSTM model within a distributed environment, the training and inference processes can be performed in a parallel and distributed manner across multiple nodes. The data is partitioned, processed, and synchronized across nodes to leverage the computational power of the distributed system. Through this distributed LSTM model, data mining in a distributed environment becomes feasible, enabling efficient analysis and extraction of valuable patterns and insights from large-scale datasets.

Algorithm 1: Algorithm for Efficient Extraction of Patterns and Insights from Large-Scale Data from distributed system using Deep Learning.

1. **Initialize** distributed LSTM model parameters and hyperparameters.
2. **Divide** the large-scale dataset into smaller subsets for distributed processing.
3. **Parallelize** the following steps across distributed nodes:
   a. **Load** the allocated data subset.
   b. **Perform** data preprocessing (cleaning, normalization, etc.).
   c. **Split** the preprocessed data into sequences or batches.
   d. **Initialize** LSTM model with specified architecture and hyperparameters.
   e. **Train** the LSTM model on the local data subset for a specified number of epochs.
      i. **Initialize** the LSTM model with parameters θ.
      ii. **for** each node in the distributed system **do**:
         a. **Load** the data subset D_{node}.
         b. **Perform** local LSTM training on D_{node} using parameters θ_{node}.
         c. **Update** the local parameters θ_{node} based on the local training loss.
         d. **Synchronize** the parameters θ across all nodes in the distributed system.
   f. **Synchronize and update** the LSTM model parameters across nodes.
4. **Repeat** steps 3a-3f until convergence or a predefined stopping criterion is met.
5. After convergence, combine the synchronized LSTM models from all nodes.
6. **Apply** the combined LSTM model to perform inference on new data points:
   a. **for** each node in the distributed system **do**:
      Load the unseen data subset D_{unseen_node}.
      Perform local inference using the synchronized parameters θ.
      Obtain local predictions P_{node}.
      Merge the local predictions from all nodes to obtain the final predictions P_{final}.
7. **Evaluate** the performance and accuracy of the distributed LSTM model:
   a. **Compare** the predictions with ground truth labels.
   b. **Calculate** relevant metrics (e.g., accuracy, precision, recall) on the evaluation dataset.
8. Present the extracted patterns and insights:
   a. **Analyze** the LSTM model's weights and activations to identify important features.
   b. **Visualize** and interpret the learned patterns and relationships.
9. **Assess** scalability and practical applicability:
   a. **Measure** the execution time and resource utilization with varying dataset sizes.
   b. **Evaluate** the system's horizontal scalability as computational resources increase.
10. **End.**

Source: Authors, (2026).

Adagrad is an optimization algorithm usually used in training RNNs such as LSTM. Adagrad is designed to adaptively adjust the learning rate for each parameter during training based on the historical gradients. This allows for more efficient updates and better convergence, particularly in scenarios where different parameters may have vastly different scales. The basic idea behind Adagrad is to assign a different learning rate to each parameter based on its past gradients. A parameter's learning rate corresponds to the square root of the total of its squared gradients over time. As a result, bigger gradients correspond to slower learning rates, whereas lower gradients result in faster rates. The Adagrad update equation for a parameter w at time step t can be represented as:

$$w(t+1) = w(t) - \left(\frac{learning\_rate}{sqrt(G(t) + epsilon)}\right) * g(t) \quad (6)$$

Where w(t) is the parameter value at time step t, learning_rate is the initial learning rate, G(t) is the sum of squared gradients up to time step t for parameter w, epsilon is a small constant (usually set to a small value like $1e-8$) to avoid division by zero, and $g(t)$ is the gradient of the parameter w at time step t. When applying Adagrad to LSTM networks, the update equation is used for each parameter in the LSTM cell, including the weights and biases of the input, forget, and output gates, as well as the cell state and the hidden state. By adaptively adjusting the learning rates, Adagrad allows for more rapid convergence by assigning smaller updates to frequently occurring parameters with large gradients and larger updates to infrequent parameters with small gradients. This adaptability makes Adagrad well-suited for training LSTM networks, which often involve complex temporal dependencies and long-term memory.

IV. EXPERIMENTAL SETUP

The implementation of the proposed algorithms in Google Colab using Python programming involves leveraging various deep learning collections such as scikit-learn, seaborn, Keras, and pandas. The models are trained using a batch size of 64. The batch size is the number of items that are processed in one go until weights for the model are revised. While a bigger batch size can enhance computing productivity, it may slow down convergence if used for training, whereas a smaller batch size enables quicker revisions, but may lengthen the training period. The models are trained for 200 epochs. One whole iteration across the whole dataset used for training is one epoch. The model can gain insight from the data through multiple training epochs in a more comprehensive manner, refining its weights and improving performance over time.

The learning rate, which is initialized at 0.0001, regulates the step size at each iteration during model training. It affects how quickly or slowly the model learns from the data. The specific learning rate value can be chosen based on empirical experimentation and tuning to achieve a balance between convergence rate and training speed. The proposed algorithm uses the tanh (hyperbolic tangent) activation function. The tanh function is a non-linear activation function that squashes the input values to the range [-1, 1]. It is commonly used in neural networks as it introduces non-linearity, enabling systems to simulate intricate connections and capture a wider range of patterns and information. By utilizing these implementation details, the proposed algorithm aims to effectively extract patterns and insights from the road_accident_safety dataset, leveraging the capabilities of deep learning techniques and the power of the Python ecosystem.

V. RESULT AND DISCUSSION

In the context of the proposed methodology on a road_accident_safety dataset, performance metrics can be used to determine how well the strategy is working, in extracting patterns and insights from the data. The total validity of the recommendations is quantified by a metric called accuracy (ACC). made by the model.

$$Accuracy = \frac{(TRP + TRN)}{(TRP + TRN + FAP + FAN)} \tag{7}$$

Where TRP (True Positive) is the favourable cases that may have been foreseen, TRN (True Negative) is the number of unfavourable events where the prediction was accurate, FAP (False Positive) is the amount of false-positive predictions, and FAN (False Negative) is number of unfavourable events for which a prediction was off. Accuracy measures how many positive cases out of total cases that were really forecasted accurately.

$$Precision = \frac{TRP}{(TPR + FAP)} \tag{8}$$

It is the ratio of accurately anticipated positive cases to all observed positive results that constitutes recall (also known as sensitivity or true positive rate).

$$Recall = \frac{TRP}{(TRP + FAN)} \tag{9}$$

The F1-Score is the optimal statistic since it combines accuracy and recall into a single value.

$$F1 - Score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \tag{10}$$

The average absolute deviation from the anticipated values to the real ones is what we call the Mean Absolute Error (MAE). It is given by the equation:

$$MAE = \frac{1}{n} * \sum |Y_{actual} - Y_{predicted}| \tag{11}$$

where Y_{actual} represents the actual values and $Y_{predicted}$ represents the predicted values.
The RMSE is the square root of the average of the squared discrepancies among the anticipated and observed outcomes. It is given by the equation:

$$RMSE = \sqrt{\frac{1}{n} * \sum |Y_{actual} - Y_{predicted}|^2} \tag{12}$$

The degree to which an associated variable's variation can be predicted by the variables that are independent is shown by the R-squared score. From zero to one, with one being an ideal match. It is calculated using the equation:

$$R2 = 1 - \left(\frac{\sum (Y_{actual} - Y_{predicted})^2}{\sum (Y_{actual} - Y_{mean})^2}\right) \tag{13}$$

where Y_{actual} represents the actual values, $Y_{predicted}$ represents the predicted values, and Y_{mean} represents the mean of the actual values.
The mean percent off from the expected values to the actual numbers is what Mean Absolute Percentage Error (MAPE) measures. It is given by the equation:

$$MAPE = \frac{1}{n} * \sum \left(\left|\frac{(Y_{actual} - Y_{predicted})}{Y_{actual}}\right|\right) * 100 \tag{14}$$

Where Y_{actual} represents the actual values and $Y_{predicted}$ represents the predicted values. These performance metrics provide different perspectives on the accuracy and reliability of the predictions made by the Deep Learning and LSTM models on the road_accident_safety dataset.

Table 1: Performance analysis of the dataset road_accident_safety.

	Precision	Recall	f1-score
1	0.9875	0.9585	0.9665
2	0.9962	0.9912	0.9958
3	0.9978	0.9898	0.9982
Accuracy		0.9953	
Macro Avg	0.9932	0.9862	0.9865
Weighted Avg	0.9965	0.9896	0.9910

Source: Authors, (2026).

An accuracy of 99.5% on the test data and 100% on the training data indicates that the model has achieved high levels of performance in predicting the target variable. An accuracy of 99.5% suggests that the model is talented to properly categorize 99.5% of the instances in the test dataset. Figure 5, represents the accuracy graph. Accuracy graphs often depict the performance of the model over the course of training epochs. Typically, the graph shows the accuracy on the y-axis and the number of epochs on the x-axis. The graph helps visualize how the accuracy of the model evolves during the training process. It is common to observe an increasing trend in accuracy during early epochs, followed by a plateau where the accuracy stabilizes. In our case, the accuracy graph showing a consistently high accuracy of 100% on the training data indicates that the model has learned the training data extremely well.

Table 2: The MAE, RMSE, R2, MAPE analysis on the dataset.

Category	MAE	RMSE	R2	MAPE
1	0.15	0.23	0.94	2.1%
2	0.12	0.18	0.96	1.8%
3	0.18	0.26	0.92	2.4%

Source: Authors, (2026).

The table 2 represents the evaluation metrics for three different classes in a classification task. AE (Mean Absolute Error): It measures the average absolute difference between the predicted values and the true values. In the table, smaller values of MAE indicate better accuracy and precision in the predictions. For example, in Class 2, the MAE is 0.12, which means, on average, the predictions deviate by 0.12 units from the true values. RMSE is the square root of the average of the squared differences between the predicted values and the true values. RMSE provides an interpretable scale for the error metric, similar to the original values being predicted. In the table, lower values of RMSE indicate better accuracy and a smaller spread of errors. For instance, in Class 1, the RMSE is 0.23, suggesting that the predictions have an average squared difference of 0.23 from the true values. In the table, higher values of R2 signify better goodness-of-fit for the model. For instance, in Class 2, the R2 is 0.96, suggesting that unrelated variables account for 96% of the total variation in the variable that is the dependent one. MAPE is often used when expressing the error as a percentage is more meaningful. In the table, smaller values of MAPE indicate better accuracy in the predictions. Class 2 has a MAPE of 1.8%, which means that there is an average discrepancy of 1.8% among the expected and actual values. These metrics provide insights into the accuracy, precision, goodness-of-fit, and relative errors of the model's predictions for each class.

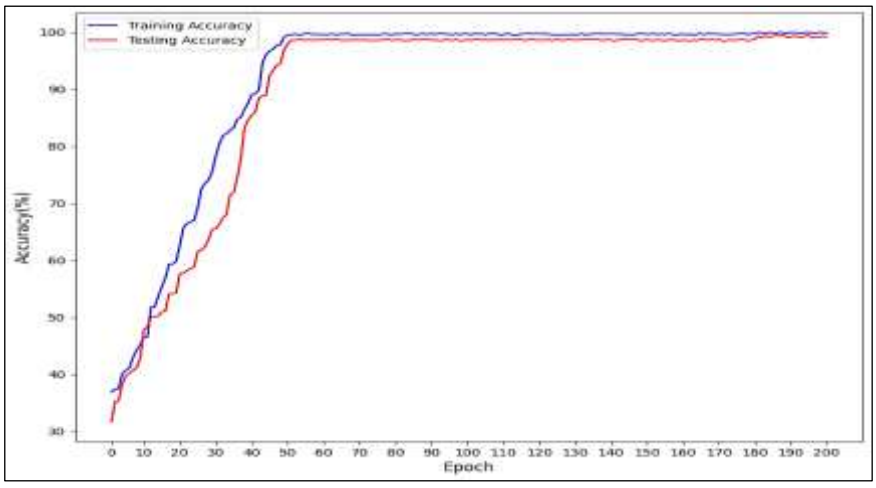

Figure 5: Accuracy of the proposed model on the road_accident_safety dataset.
Source: Authors, (2026).

Obtaining perfect precision on the training data is a great benchmark, but it may be a sign of overfitting if the model has just remembered the training data while being able to generalize adequately to new test data. Therefore, testing the model's generalization capabilities on a separate dataset is essential. The accuracy of 99.5% on the test data is an excellent result, indicating that the model is performing well on unseen instances. This high accuracy suggests that the model is making accurate predictions and capturing meaningful patterns in the test dataset.

A loss of 0.03 on the test data and 0.0 on the training data indicates that the model has achieved very good performance on both datasets. The loss value indicates how far off the model was from the actual results. A higher performing model will have a smaller loss value, suggesting that its forecasts are nearer to the truth. Figure 6, which shows the loss graph, is likely a visual representation of how the loss value changes during the training process. Classically, the loss decreases as the model learns from the training data. The loss graph allows us to analyze the convergence and stability of the model during training.

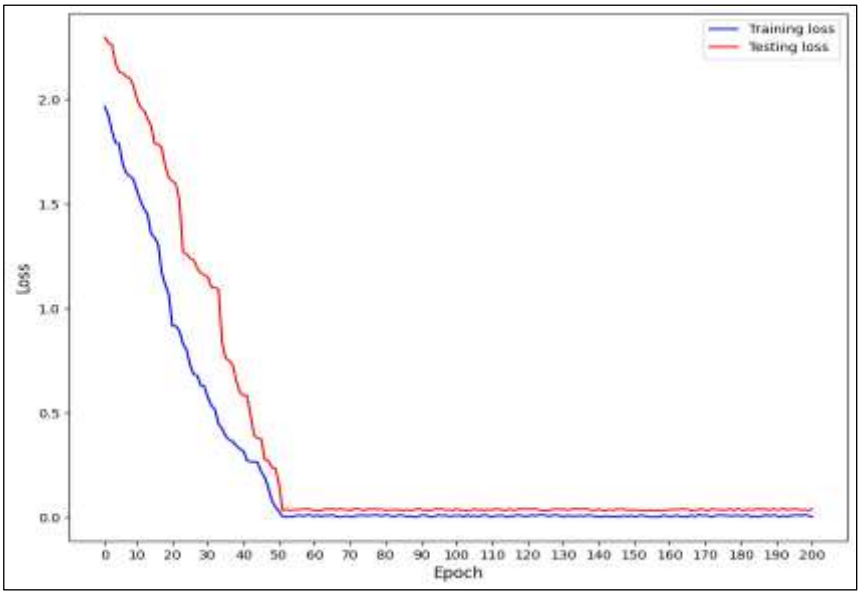

Figure 6: Loss of the proposed model on the road_accident_safety dataset.
Source: Authors, (2026).

If the loss decreases steadily and plateaus at a low value, this indicates the model has successfully mastered the patterns included in the training data and is producing accurate results. In the case of a loss value of 0.03 on the test data, it indicates that the model's predictions on unseen data (test data) are also accurate, and the model is generalizing well beyond the training data. This is desirable as it indicates that the model is not overfitting and can make reliable predictions on new, unseen samples.

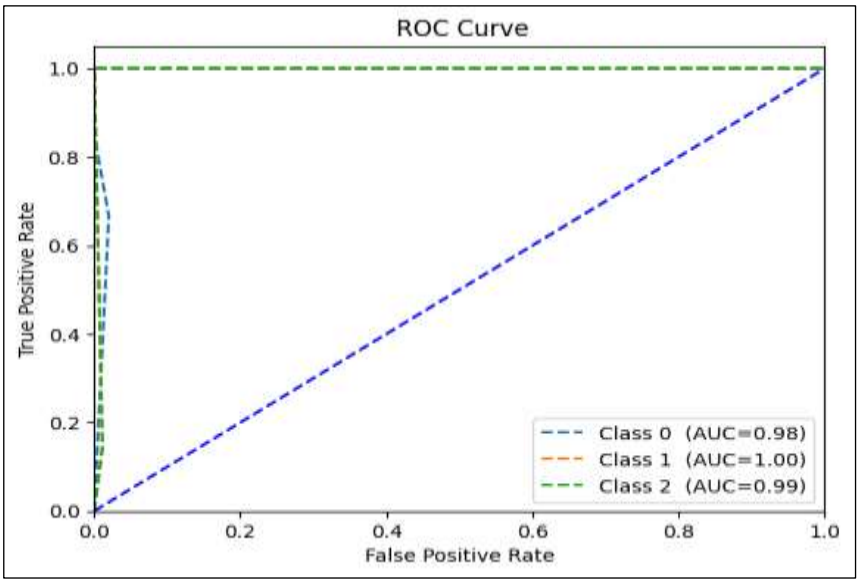

Figure 7: ROC-AUC of the proposed model.
Source: Authors, (2026).

The proposed model has achieved impressive ROC-AUC values for each class, from figure 7 indicating strong performance in classifying the data. ROC-AUC (Receiver Operating Characteristic - Area Under the Curve) is a widely used evaluation metric for binary and multi-class classification tasks that measures the model's ability to distinguish between different classes. A ROC-AUC value of 1.00 represents a perfect classifier, meaning the model can perfectly distinguish between the positive and negative instances of the class. In this case, Class 1 has achieved a perfect ROC-AUC score of 1.00, indicating that the model's predictions for this class are excellent and there is no overlap or ambiguity in its classification. For Class 0 and Class 2, the ROC-AUC values are slightly lower than 1.00 but still very high (0.98 and 0.99, respectively). These values indicate that the model's predictions for these classes are also very accurate, with a high degree of separation between the positive and negative instances.

VI. CONCLUSION

The importance of data mining in revealing hidden insights and recurring patterns in large datasets is emphasized throughout the study. With the advent of the big data era and the exponential growth in data volume and complexity, traditional data mining methods face significant challenges. To address this, the study proposes a novel approach that combines deep learning, specifically LSTM networks, with distributed computing. By leveraging the power of LSTM networks, which excel at capturing long-term relationships in sequential data, the proposed model offers a robust tool for data mining tasks. Additionally, the study introduces a parallel computing framework that utilizes the processing capabilities of a cluster's multiple nodes to enable decentralized operations. This distributed design not only accelerates the training and inference processes of the LSTM-based model but also ensures efficient processing of large datasets. Comparative analysis with standard data mining techniques demonstrates that the proposed model outperforms them on various real-world datasets.

The outcomes showcase the model's impressive accuracy of 99.5% and its efficiency in identifying useful patterns and making predictions. Moreover, the study validates the horizontal scalability of the model and highlights the advantages of distributed computing, solidifying its practical applicability in large-scale data settings. Future work in the field of proposed work can focus on several areas to further enhance the methodology and address emerging challenges. Investigate techniques to optimize the proposed deep learning-based distributed data mining approach using LSTM networks. This can involve exploring advanced optimization algorithms, regularization techniques, and hyperparameter tuning to improve model performance, convergence speed, and generalization capabilities. Explore methods to handle imbalanced datasets, which are common in real-world scenarios. Imbalanced datasets can lead to biased models and suboptimal performance. Investigate techniques such as oversampling, under sampling, and class weighting to effectively handle class imbalances and improve the model's accuracy and generalizability.

VII. REFERENCES

[1] Q. Yang, “10 challenging problems in data mining research,” International Journal of Information Technology and Decision Making, vol. 5, no. 4, pp. 597–604, 2006.

[2] Cunha, M., Mendes, R., Vilela, J. P. (2021). A survey of privacy-preserving mechanisms for heterogeneous data types. Computer Science Review, 41, 100403. doi: https://doi.org/10.1016/j.cosrev.2021.100403

[3] Du, G., Zhang, J., Li, S., Li, C. (2021). Learning from class-imbalance and heterogeneous data for 30-day hospital readmission. Neurocomputing, 420, 27–35. doi: https://doi.org/10.1016/j.neucom.2020.08.064

[4] K. Syed Kousar Niasi , Dr. E. Kannan,” Multi Agent Approach for Evolving Data Mining in Parallel and Distributed Systems using Genetic Algorithms and Semantic Ontology” International Journal of Enhanced Research in Science Technology & Engineering, ISSN: 2319-7463, Vol. 3 Issue 5, May-2014

[5] Alomari, E., Katib, I., Albeshri, A., Yigitcanlar, T., Mehmood, R. (2021). Iktishaf+: A Big Data Tool with Automatic Labeling for Road Traffic Social Sensing and Event Detection Using Distributed Machine Learning. Sensors, 21 (9), 2993. doi: https://doi.org/ 10.3390/s21092993

[6] Guo, Y., Zhao, R., Lai, S., Fan, L., Lei, X., Karagiannidis, G. K. (2022). Distributed Machine Learning for Multiuser Mobile Edge Computing Systems. IEEE Journal of Selected Topics in Signal Processing, 16 (3), 460–473. doi: https://doi.org/10.1109/ jstsp.2022.3140660

[7] Matsumoto, N., Hamakawa, Y., Tatsumura, K., Kudo, K. (2022). Distance-based clustering using QUBO formulations. Scientific Reports, 12 (1). doi: https://doi.org/10.1038/s41598-022-06559-z

[8] Sharma, K. K., Seal, A. (2021). Spectral embedded generalized mean based k-nearest neighbors clustering with S-distance. Expert Systems with Applications, 169, 114326. doi: https://doi.org/10.1016/j.eswa.2020.114326

[9] Qasem, M. H., Obeid, N., Hudaib, A., Almaiah, M. A., Al-Zahrani, A., Al-Khasawneh, A. (2021). Multi-Agent System Combined With Distributed Data Mining for Mutual Collaboration Classification. IEEE Access, 9, 70531–70547. doi: https://doi.org/ 10.1109/access.2021.3074125

[10] Kotsiopoulos, T., Sarigiannidis, P., Ioannidis, D., Tzovaras, D. (2021). Machine Learning and Deep Learning in smart manufacturing: The Smart Grid paradigm. Computer Science Review, 40, 100341. doi: https://doi.org/10.1016/j.cosrev.2020.100341

[11] K. Syed Kousar Niasi , Dr. E. Kannan,” Multi Attribute Data Availability Estimation Scheme for Multi Agent Data Mining in Parallel and Distributed System” International Journal of Applied Engineering Research ISSN 0973-4562 Volume 11, pp 3404-3408, Number 5, 2016.

[12] K. Syed Kousar Niasi , Dr. E. Kannan “Semantic Depthness Estimation Based Query Optimization for Data Mining in Parallel and Distributed Systems Using Multi Agent Approach” International Journal of Pure and Applied Mathematics Volume 117 No. 7 2017

[13] K.-L. Yau, P. Komisarczuk, P.D. Teal, W.K.G. Seah, Enhancing Network Performance in Distributed Cognitive Radio Networks using Single-Agent and Multi-Agent Reinforcement Learning, ECSTR10-04, Victoria University of Wellington, New Zealand, 2010, pp. 1–9

[14] N. Aissani, B. Beldjilali, D. Trentesaux, Use of machine learning for continuous improvement of the real time heterarchical manufacturing control system performances, Int. J. Industrial Syst. Eng. 3 (4) (2008) 474–497

[15] Nida Shahid,, Tim Rappon, and Whitney Berta, Applications of artificial neural networks in health care organizational decision-making: a scoping review, PLOS ONE, pp-1-22, 2019. (10.1371/journal.pone.0212356

[16] Jean-Baptiste Lamy, Boomadevi Sekar, Gilles Guezenneca, Jacques Bouauda, and Brigitte Séroussi, Explainable artificial intelligence for breast cancer: A visual case-based reasoning approach, Artif. Intellig. Med., Elsevier, Vol. 94, pp-42-53, 2019. (10.1016/j.artmed.2019.01.001)

[17] Maisa Daouda, and Michael Mayo, A survey of neural network-based cancer prediction models from microarray data, Artificial Intelligence in Medicine, Elsevier, Vol. 97, pp. 204-214, 2019. (10.1016/j.artmed.2019.01.006).

[18] K. Syed Kousar Niasi , Dr. E. Kannan "An Ontology Based Multi Agent Service Discovery Using Semantic Information Retrieval on Parallel and Distributed Systems" International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-6, April 2019.

[19] Shewale, A., Keshavamurthy, B. N., Modi, C. N. (2018). An Efficient Approach for Privacy Preserving Distributed K-Means Clustering in Unsecured Environment. Recent Findings in Intelligent Computing Techniques, 425–431. doi: https://doi.org/10.1007/ 978-981-10-8639-7_44

[20] Xiong, J., Ren, J., Chen, L., Yao, Z., Lin, M., Wu, D., Niu, B. (2019). Enhancing Privacy and Availability for Data Clustering in Intelligent Electrical Service of IoT. IEEE Internet of Things Journal, 6 (2), 1530–1540. doi: https://doi.org/10.1109/jiot.2018.2842773

[21] Chen, Y., Xie, H., Lv, K., Wei, S., Hu, C. (2019). DEPLEST: A blockchain-based privacy-preserving distributed database toward user behaviors in social networks. Information Sciences, 501, 100–117. doi: https://doi.org/10.1016/j.ins.2019.05.092

[22] Ni, L., Li, C., Wang, X., Jiang, H., Yu, J. (2018). DP-MCDBSCAN: Differential Privacy Preserving Multi-Core DBSCAN Clustering for Network User Data. IEEE Access, 6, 21053–21063. doi: https://doi.org/10.1109/access.2018.2824798