International Journal on
"Technical and Physical Problems of Engineering"
(IJTPE)
Published by International Organization of IOTPE

ISSN 2077-3528
IJTPE Journal
www.iotpe.com
ijtpe@iotpe.com

September 2024 | Issue 60 | Volume 16 | Number 3 | Pages 350-360

A FOG-BASED FRAMEWORK FOR DETECTING MALICIOUS NODES FOR ENHANCING SECURITY IN SMART CITIES

A.K. Mehta [1] M.P. Patel [2]

1. Department of Computer/IT Engineering, Gujarat Technological University, Ahmedabad, Gujarat, India
akash.mehta.it@gmail.com
2. Department of Computer Engineering, Devang Patel Institute of Advance Technology and Research, Charotar University of Science and Technology,Gujarat, India, minalpatel.dce@charusat.ac.in

Abstract- This research starts with comparing Fog and Cloud to decide whether we should add fog as a layer to the cloud to detect malicious node in smart city applications. In our smart parking system, we employed two approaches to detect latency and malicious nodes: Cloud Computing and Fog Computing. Our observations indicate that integrating a Fog Computing layer with the cloud offers several advantages. Firstly, it addresses latency issues, ensuring faster response times for data processing. Additionally, Fog Computing provides significant improvements like improved on demand scaling, network usage, enhanced security measures, and resourcefulness progress. By combining Cloud Computing with Fog Computing, we can leverage the strengths of both paradigms to optimize system performance and mitigate potential security threats effectively. This research looks at how well Ant Colony Optimization (ACO) works in identifying malicious nodes in a smart parking system located in a smart city. K-means and Hierarchical clustering techniques are compared. The accuracy rates of ACO, K-means, and Hierarchical clustering, which are 97%, 92%, and 93%, respectively, serve as the foundation for the study. Using its precision-driven methodology, the study focuses on enhancing ACO for anomaly detection in smart parking systems, effectively identifying malicious nodes while reducing false positives. The findings suggest that ACO performs better than conventional clustering algorithms in the context of smart city security, highlighting its potential to improve security protocols and guarantee the dependability of smart parking systems.

Keywords: Malicious Node Detection, Smart City, Ant Colony Optimization (ACO), Anomaly Detection, Smart Parking Systems, Fog-Computing, Cloud-Computing.

1. INTRODUCTION

The term "fog computing" describes practice of extending cloud computing to an organization's network edge and the term was used by Cisco. This practice is also known as "edge computing" or fogging [1]. For end devices and computer data centers, this makes networking, storage, and computing services easier [2]. Fog nodes form a crucial part of the fog setup, positioned between the cloud and physical hosts. They offer data, functions, computational power, and storing predominantly for benefit of the host [3]. By handling the data nearer to the source, fog computing enhances speed, security, and system efficiency [4]. "Faulty nodes" are the nodes which operates on irregular basis because of malfunctions or accidents, and "malicious nodes" are the nodes which are compromised [5], posing threats like fake data injection (FDI), distributed denial of service (DDoS), and other attacks [6].

Nodes with stagnant trust values over time are deemed malicious, prompting source nodes to avoid using their routes to maintain security for packet transmission [7]. The mischievous behaviors of nodes like falsely recording traffic collisions, can have critical effects, which may impact drivers and safety of the traffic [8]. Identifying inner antagonistic nodes, specifically if recognized nodes turn malicious and manipulate with data, presents challenges for cryptographical methods [9]. Fog computing servers which are positioned at the edge of the network setup in fog computing, look like trivial cloud servers [10]. They provide processing, data storing, and communication assistance, assisting immediate communication involving fog devices and servers.

2. BACKGROUND

2.1. K-Means Clustering

A basic description of the conventional this section provides the k-means algorithm. Large data sets are often clustered using the well-liked data mining clustering method K-means [11]. In 1967, MacQueen introduced the k-means method for the first time [12], It was one of the most straightforward unsupervised learning techniques used to the issue involving the well-known cluster. The specified date objects are divided into k distinct clusters using this iterative partitioning and clustering approach, which eventually converges to local minimum. We will receive independent and compact output of the clusters due to this. There are two separate phases available in the algorithm. In the first stage, k centers are chosen at random; k is a predetermined number.

Registered: 240322 | Article Info: 1878 | Accepted: 240829
Reserved: 240607 | Available: 240930

After that, it is required to deliver every data item to the closest center. The distance between the cluster centers and every data point is often calculated using the Euclidean distance. Early grouping and the first stage are finished when each data item belongs to a few clusters. Recalculating the early-formed clusters' average. The criteria function is iterated through until the lowest value is obtained [13]. The criteria function is defined as follows, considering that x is the target item and that X_i is the cluster C_i average, which is shown in Equation (1) [13]:

$$E = \sum_{i=1}^{K} \sum_{X \in C_i} \left| x - x_i \right|^2 \quad (1)$$

For each database entry, E is the squared error total. The Euclidean distance is the length of the criteria function, which finds the shortest path between the cluster center and each data point. The Euclidean separation of a single vector X and another vector Y, where, $X = (X_1, X_2, \ldots X_n)$ and $Y = (Y_1, Y_2, \ldots Y_n)$. The Euclidean gap (distance) is calculated through Equation (2) [13].

$$d(X_i, Y_i) = \left[\sum_{i=1}^{n} (x_i - y_i)^2 \right]^{1/2} \quad (2)$$

The algorithm of the K-Means works as per below steps:

- Input: K will be the total number of necessary groups (Clusters), and a database D Including n items.

Algorithm 1. K- Means Standard Algorithm

1: # Initialisation
2: Value of $N := \{x_1, ..., x_n\}$;
3: Value of $M := \{\mu_1, ..., \mu_k\}$;
4: # Categorization
5: For $x_i \in N$ and $\mu k \in M$
6: Euclidean gap from every x_i to k Clustering Calculation
7: Designate object x_i to nearest Centroid μk
8: Calculation for Centroid:
9: Calculate Centroid μk
10: # Convergence:
11: If $M := \{\mu l, ..., \mu k\}$ remains unaffected in Two successive iterations
then:
12: End the Procedure;
13: else
14: Go to Categorization
15: End

2.2. Hierarchical Clustering

Organizing optimization techniques according on how many clusters there are initially, before the clustering. In contrast, hierarchical clustering algorithms merge or split pre-existing groupings and define the sequence in which clusters are split apart or joined together. Hierarchical clusters may be shown using a tree or dendrogram. Two methods are available for achieving hierarchical clustering. Both top-down and bottom-up approaches are possible. Big clusters split up into smaller clusters, while little clusters inside larger clusters merge together [14]. The hierarchical technique is further classified as follows:

2.2.1. Divisive Hierarchical Clustering

As a "reverse" kind of agglomerative clustering, "divisive clustering" divides the data points recursively from a starting point of one cluster or model. The process is carried out again until a predetermined number K of clusters or models -a stopping criterion- is satisfied. After every division iteration, the items in the "poorest-fit" cluster have the lowest probability of being divided. Until the clusters become singletons or a stop requirement is satisfied, this procedure is repeated. Similar to agglomerative clustering, this has problems with model selection and high processing costs [15]. Furthermore, since the data may initially be divided into two clusters, it is very sensitive to initialization.

To create divide (top-down) grouping, follow these steps:

- Step 1: start with each cluster's data point.
- Step 2: Remove the "outsiders" from the least cohesive group at the conclusion of every cycle.
- Step 3: Step 2 should be followed if every example is not in a singleton cluster.

2.2.2. Hierarchical Agglomerative Clustering (HAC)

A bottom-up method where every element represents a cluster that is successively combined to create the desired cluster structure. There are N clusters at the beginning of this N-sample method, and each cluster has one sample. Then, until there are no more clusters or the user designates, the two clusters that are most similar will merge. The algorithm's parameters are the center, average, minimum, and maximum distances [16].

Agglomerative (bottom-up) clustering is formed by the following steps:

- Step 1: To start with, treat every data point as a separate singleton cluster.
- Step 2: Merge the two clusters that have the minimum distance after each Euclidian distance calculation iteration.
- Step 3: Proceed to step 2 if there isn't a single cluster containing every example.

Algorithm 2. Hierarchical Agglomerative Clustering (HAC)

1: Input: Data vectors $\{x_n\}_{n=1}^{N}$, distance groupwise DIST $\left(G, G^1\right)$
2: $A \leftarrow \theta$ ▷ Operational set establishes out blank.
3: for $n \leftarrow 1...N$ do ▷ Loop over the Data.
4: $A \leftarrow A \cup \{\{x_n\}\}$ ▷ Add each data point as its own cluster.
5: end for
6: $T \leftarrow A$ ▷ Save the hierarchy as an order of mixes. In system, pointers.
7: while $|A| > 1$ do ▷ Loop up until the operational set only has one data item.
8: $G_1^*, G_2^* \leftarrow$ arg min DIST (G_1, G_2) ▷ Select pair in A with best gap. $G_1, G_2 \in A; G_1, G_2 \in A$
9: $A \leftarrow \left(A \setminus \{G_1^*\}\right) \setminus \{G_2^*\}$ ▷ Remove each from active set.
10: $A \leftarrow A \cup \{G_1^* \cup G_2^*\}$ ▷ Add union to active set.
11: $T \leftarrow T \cup \{G_1^* \cup G_2^*\}$ ▷ Add union to tree.
12: end while
13: Return: Tree T.

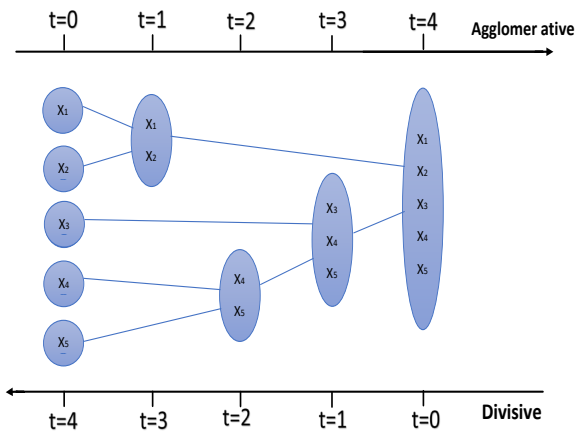


Figure 1. Hierarchical Clustering [16]

2.3. Ant Colony Optimization

A metaheuristic based on population for resolving optimization issues was first introduced as ant colony optimization [17]. ACO constructs solutions piecemeal using artificial ants, choosing parts according on heuristic understanding of the problem and pheromone pathways that indicate the acquired expertise in conducting searches. In method 1, a single objective function is minimized in a combinatorial optimization problem, demonstrating the basis of an ACO technique. The ACO first determines the starting values of the pheromone trail (T) and the solution components' heuristic value (H, or visibility). Until a predefined stop condition is satisfied, the application keeps running. Every iteration step consists of four stages. In the initial phase, each ant within the colony independently constructs a solution in an autonomous, asynchronous, and concurrent manner. This process involves selecting components based on a probabilistic rule that takes into account two key factors:

Experience gained from previous searches: This is represented by the trace of pheromone deposited by other ants. Ants are more likely to select components that have been previously visited and have higher pheromone concentrations, indicating successful paths. Heuristic information about the components: This is determined by the visibility of components. Ants assess the desirability of selecting a particular component based on its heuristic information, which may include factors such as proximity, attractiveness, or other relevant characteristics.

By integrating these two factors into their decision-making process, ants autonomously navigate the search space, asynchronously constructing solutions concurrently with other ants in the colony. This collective exploration strategy allows the colony to efficiently search for optimal solutions to the given problem. To enhance the solutions, a local search approach is optionally used in the subsequent step. During the third phase, the pheromone trails undergo an update, and the net change in the pheromone value is ascertained by summing together the contributions from the two update methods. The trace readings are increased by pheromone deposits in the solution's component components while they are decreased by evaporation.

Finally, the best answer found so far 'also referred to as the best-so-far solution' is adjusted in the last stage if a better one is found. When the stop criteria are met, ACO provides excellent results observed so far.

Algorithm 3. Ant Colony Optimization (ACO)

Require: Instance of *CO* challenge model P
where $P = (S, f, \tau)$ Set Pheromone Values (τ)
sbs Value← NULL
while execution finish state not seen do
Siter Value ← φ
for $j = 1$ to n_a do
$s \leftarrow$ Assemble Resolution (τ)
if s is an appropriate result, then
$s \leftarrow$ Local Exploration(s) {optional}
if ($f(s) < f(sbs)$) or (sbs = NULL) then
sbs Value ← s
end if end if
Siter Value ← *Siter* $\cup \{s\}$
end for
Apply Pheromone Update (τ, *Siter*, *sbs*)
end while

- In any conventional method employing the random proportionate state transition algorithm, pheromone deposition occurs based on the quality of solutions found by all ants. This means that all ants contribute to pheromone deposition, and the quantity of pheromone accumulated depends on the superiority of the resolutions they retrieve. Additionally, pheromone evaporates from all parts of the Ant System (AS), ensuring that older pheromone trails gradually diminish over time.
- On the other hand, in the Ant Colony System (ACS) utilizing the pseudo-random state transition method, only elements belonging to the optimal solution receive pheromone deposition. This selective pheromone deposition strategy helps reinforce the paths leading to high-quality solutions.

Moreover, ACS integrates a local pheromone update mechanism into the solution development process, allowing ants to explore underutilized components more effectively. This local pheromone update enhances the exploration. The MAX − MIN Ant System (MMAS) only deposits the pheromone on the elements of the optimal solution and has explicit lower and upper bounds on it. Additionally, ACO techniques have been suggested as solutions for dynamic and multi-objective optimization issues.

2.4. Contribution of the Study

The study on "Malicious Node Detection Using Fog Computing in Smart City" introduces a novel approach for detecting malicious nodes within smart city infrastructure using fog computing. By progressing cloud to the edge of the system network, fog computing provides a decentralized architecture appropriate for real-time security threat detection and response. This approach enhances security by detecting and mitigating threats on the edge of the network, ensuring the integrity and reliability of critical services. It focuses on real-time detection, minimizing the impact of malicious activities. The approach offers scalability, adapting to the dynamic nature of smart city environments.

A distributed architecture that can extend horizontally to handle the increasing number of devices and data in smart city networks is offered by fog computing. It also emphasizes resource efficiency by offloading security tasks to edge devices within the fog computing infrastructure. The study enhances the resilience of smart city infrastructure against cyber-attacks and security breaches by proactively identifying and neutralizing threats. This approach lays the foundation for building more secure and resilient smart cities of the future.

2.5. Organization of the Paper

- Introduction: This introduces the research topic, outlining the significance of detecting malicious nodes in smart city environments. In Literature Review, previous studies related to malicious node detection in various network types are reviewed.
- Methodology: The methodology outlines the experimental setup and procedures followed in the study.
- Results: This presents the experiment results conducted, focusing on the comparative analysis of the algorithms' performance.
- Discussion: The discussion interprets and analyses the results obtained, discussing the implications of ACO's effectiveness in detecting malicious nodes.
- Conclusion and Future Work: Finally, the conclusion summarizes the key findings of the study.

3. LITERATURE REVIEW

Numerous research efforts have been motivated by the growing emphasis on malicious node identification in different kinds of networks. In MANETs, Wang, et al. devised a detection method grounded in trust, leveraging perceptions like conviction variability and indication sequence [18]. Their methodology evaluates nodes' faith estimates over time, employing conviction variability to focus substantial instability and indication sequence to pinpoint malicious activities.

Similarly, Ebinger, et al. [19] suggested a collaborative interference detection method for MANETs established on repute conversation and trust measurement. They classify credit data info into substituted repute and conviction, merging them to improve interference recognition capabilities. For Mobile Wireless Sensor Networks (MWSNs), several trust administration procedures have been suggested, mainly concentrating on data truthfulness, assured transmitting, and network precautions. Shaikha et al. [20] established a trust administration arrangement based on group, with a focus on directly perceived Quality of Service (QoS) system of measurement.

Additionally, exhibited an effective system competent of identifying malicious nodes, distinguishing that equally contingent and independent dangerous node influence the object discovery in WSNs [21]. In the domain of MWSNs, Abdelhakim, et al. suggested a strong method to detect malicious nodes in the existence of Byzantine incidents, highlighting adaptive data mixture in active threat surroundings [22]. Their technique operates a consortium model established on randomness for investigation.

Furthermore, Vempaty, et al. [23] established two distinct approaches to neutralize Byzantine confrontations in MWSNs. One approach occupies Byzantine node documentation imagining undeviating neighborhood quantizers, while the other mixes this detection scheme with dynamical non-uniform tolerance quantization across sensors. Proposes a nasty vehicle recognition representation based on spatio-temporal features of traffic flow under cloud-fog computing-based IoVs [24]. It divides urban road networks into multiple traffic subareas, extracts spatial and temporal features, and uses a short-term prediction model. Experimental results show the scheme is effective and efficient in detecting malicious vehicles.

According to [25] the IoT is crucial for Smart cities, with increasing data demand and latency. Fog computing helps reduce latency by providing storage, computing, and networking services. It's essential for smart IoT infrastructure and waste management. Open challenges and directions for future research are discussed. [26] aims to provide a novel method known as the Assertive Trust Based Efficient Technique to detect and disregard reprobate points from edge grounded systems for smart cities. The method increases latency, reliability, and network life span by using penalized trust and packet loss rate [27]. Smart city residents can benefit from real-time services through collaborative applications, but latency-aware network services are needed.

Technological enhancements are needed to protect against identity theft and misuse. EBAD, an Edge-based approach, is designed to identify Sybil attacker nodes and assess malicious behavior [28]. Presents a method for protecting IoT devices in smart cities through attack and machine learning methods for detecting anomalies. It investigates feature selection, ensemble approaches, multi-class classification and cross-validation. According to experimental findings, the method successfully detects cyberattacks.

4. RESEARCH GAP

There is a gap in the research for the study "Malicious Node Detection using Fog Computing in Smart City" when it comes to using certain detection methods, like Ant Colony Optimization (ACO) in smart parking systems module of any smart city. Sufficient research is already done to find out the Malicious nodes in different types of networks like MANETs or MWSNs but still not too much work is done using Ant Colony Optimization to find our Malicious Nodes in Smart Parking Systems module of Smart City. Also, comparing ACO to clustering algorithms like K-means and hierarchical clustering is useful for figuring out how well it works in this particular application area. So, there aren't many studies that look at how to improve security and make sure that smart parking systems in smart cities work reliably. These studies would help by comparing and optimizing ACO with clustering algorithms that are designed for these kinds of systems.

4.1. Data Collection

For this study we generated the data by running the traffic classifier file, to categorize and track different network traffics. The traffic classifier file processed network data into four distinct CSV files; "telnet", "DNS", "ping". Each of these files provides useful information about activities on a given network such as telnet sessions, DNS lookups, ping replies and voice call transmissions. These CSV datasets are a springboard for our research enabling us to interrogate many dimensions of network behavior and performance in smart parking systems. By closely studying these datasets, we were able to draw out important findings and insights into how fog computing architectures and optimization methodologies can be used in enhancing smart vehicle parking systems.

1. DNS: DNS data provides knowledge about what goes on during network domain name resolution. It is useful for analyzing communication patterns, finding potential problems in the system and ascertaining the efficiency of smart parking DNS resolution.
2. Ping: Ping data gives a view on network latency and reachability. This will enable researchers to evaluate network latency performance by looking at ping responses, identify issues related to possible network congestion as well as make a comparison between latencies in different architectures or segments of networks
3. Telnet: Telnet data shows details about interactive sessions with remote computers that were established. Even though telnet is less likely used in modern day systems due to security reasons, it can still be employed to analyze remote access activities, mitigate potential risks and evaluate the overall network health in the smart parking system.

5. SYSTEM ARCHITECTURE

A framework for a malicious node detection system under cloud and fog computing based IoTs that is utilized to extract all the aspects of traffic network. As shown in Figure 1, our proposed structure is constituted with three layers and they are fog, clouds, and the physical layer. Each layer is optimized and equipped for smart parking and smart manufacturing environments.

5.1. Physical Layer

In this layer, the key components are sensors, and vehicles they further are categorized into two types malicious and non-malicious entities. In reference, the sensors will collect and transmit the data which is related to parking space availability, duration and vehicle characteristics. And also, the sensors in smart manufacturing will look after the equipment status, environmental conditions and production metrics.

5.2. Fog Layer

The fog layer consists of fog and edge nodes which are deployed on various locations of the network. In smart parking the parking sensors interacts with the fog nodes in order to acquire real time occupancy data and manage the allocation process efficiently in the same way these fog nodes interact with the industrial, domestic sensors in order to monitor machine performance and detect abnormalities and process optimization.

5.3. Cloud Layer

The data collected from the fog layers is analyzed in a centralized cloud server, when it comes to smart parking these cloud servers are responsible for analyzing the data and provide all the insights of the utilization patterns of the parking and facilitate predictive management of smart parking.

5.4. The Key Components

- Sensors: These are the key elements which are responsible to collect and transmit the relevant data like status of the available parking slots, environmental factors and status of the equipment's.
- Detector: Detectors are responsible to detect events such as entry and exit of vehicles and occupancy status.
- Fog Node: Fog nodes are in charge of processing and analyzing data more quickly and making decisions in real time along with these fog nodes are responsible for such as local data aggregation and anomaly detection.
- Cloud Server: Cloud servers are the units with centralized data processing and analytics capabilities. They can perform advanced and complex algorithms on aggregated data to extract valuable insights and able to identify malicious behaviors of the nodes across the smart parking ecosystem.

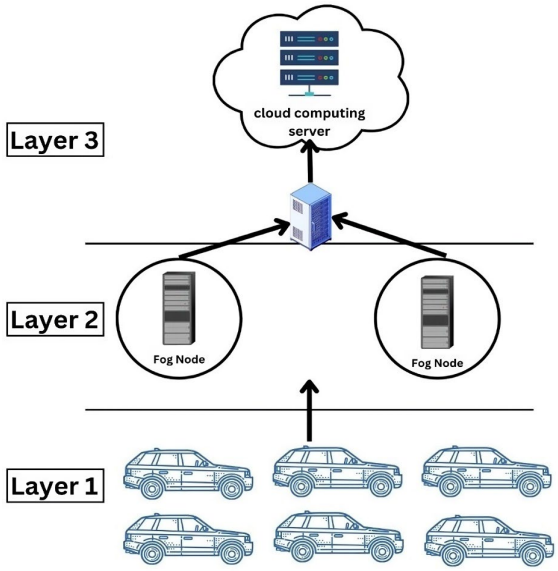


Figure 2. Malicious vehicle detection workflow [29]

5.5. Attack Threat

Smart parking systems are associated potential threat of malicious activities which effects the system and misleads the drivers and disturb parking activities these malicious vehicles tend to provide false information of occupancy to the smart parking management system. Identifying such activities are challenging when the malicious actors have valid credentials or access of the system.

5.6. Detection System

Our proposed detection system identifies vehicles with malicious behaviors by utilizing the data of occupancy, vehicle count and movement, duration of stay, payment transactions, environmental data, access control data, traffic data and location data etc. The detection scheme involves several steps depicted in Algorithm 1.

1. Dividing the parking slots into various sub zones depends upon the GPS data provided. Each fog device monitors specific subzone and acquires data from the sensors regarding occupancy and all. Fog devices utilize this data to analyses and identify available parking slots.
2. The aggrade data from the fog devices are transmitted to the cloud servers in order to identify malicious vehicles.
3. Detection of anomalous parking behavior depends upon the deviation of expected and observed occupancy patterns.

5.7. Problem Formulation

Using the K-Means approach, we seek to separate a dataset consisting of N information Data objects into K number of groupings. To designate the sharing of data point X_i to cluster k, we introduce binary variables $\delta_{i,k}$. We also propose binary variables, β_i, to detect abnormalities. This is how the objective function J is expressed:

$$J = \sum_{i=1}^{N} \sum_{k=1}^{K} \delta_{i,k} . d(x_i, \mu_k)^2 + \alpha \sum_{i=1}^{N} \beta_i \quad (3)$$

The gap between the data points X_i and the centroid μ_k is denoted by $d(x_i, \mu_k)$, where α is a hyper parameter that regulates the balance between anomaly detection and clustering accuracy.

5.8. Optimization Problem

The definition of the optimization is:

Minimize J with respect to $\{\mu_k\}, \{\delta_{i,k}\}, \{\beta_i\}$ Subject to:

1. One cluster is allocated to each data point: $\sum_{k=1}^{K} \delta_{i,k} = 1$ for all i.
2. Anomalies are correctly identified: $\beta_i = 1$ if x_i is an anomaly.
3. No cluster is identified to anomalies: $\sum_{k=1}^{K} \delta_{i,k} \leq 1$ for all i.

By penalizing anomalies in addition to capturing regular patterns, this optimization approach makes sure that the clustering process produces a more reliable anomaly detection model.

5.9. Performance Metrics

Algorithms are measured using performance metrics derived such as the F1-score, sensitivity, accuracy, and precision, from the confusion matrix [30].

- Accuracy: The proportion of properly identified subjects to the total number of subjects is recognized as the subject recognition measure.

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (4)$$

- Sensitivity: The ratio of true positive labels that our system can accurately identify is called recall; for our purposes, recall is the same as sensitivity.

$$Sensitivity = TP / (TP + FN) \quad (5)$$

- Precision: An outlook's accuracy may be determined by counting the number of predictions that are correct. Another term for this concept is "predictive value".

$$Precision = TP / (TP + FP) \quad (6)$$

- Specificity: In particular, the technique has identified the drawback.

$$Specificity = TN / (TN + FP) \quad (7)$$

where, FP is False Positive, FN is False Negative, TP is True Positive, TN is True Negative.

6. RESULTS AND DISCUSSION

The findings section offers a brief summary of the performance metrics and results attained through examining fog computing architectures and anomaly detection models in smart vehicle parking systems. This enables more complete understanding of how latency, network utilization rates, accuracy levels, precision rates among others can be combined with F1 score curves to predict which methodologies work best. These findings in turn lay the groundwork for informed conversation as well as ramifications later on.

Table 1. Smart car parking for cloud and fog computing: simulation results for different numbers of cameras

No. of Cameras	Latency Value of Fog	Latency Value of Cloud	Fog: Usage of Network in *KB*	Cloud: Usage of Network in *KB*
16	8.69	7.91	27342	3321.4
20	9.23	8.02	36443	4002.3
24	10.53	8.41	43789	4657
28	12.13	7.57	55102	5602
32	282.3	8.4	62108	6257
40	1890.7	9.37	64202	7908.5
44	2402	10.23	65102	8702
48	2799	10.26	65023	9147.3

From the above table it is observable that the Comparison of Fog computing with Cloud computing system. Real-time data analysis should be done using our suggested fog computing architecture rather than cloud-based solutions because processing is decentralized and closer to the data source where smart parking sensors usually have lower latency. For example, fog latency ranges from 7.53 to 10.32 ms depending on how the sensor is set up, indicating a higher reaction time of smart parking sensors.

On the other hand, unlike cloud latency that varies more with low predictability for inefficiently remote processed information, there is a steadier rise in network utilization as instances grow due to additional sensor deployment but at much lower rates compared to what happens with cloud-based scenarios thus showing lower demand on network bandwidths in comparison with them. Smart parking systems can be optimized effectively by utilizing fog computing better with less latency as well as network efficiency at the edge of the network than cloud computing does.

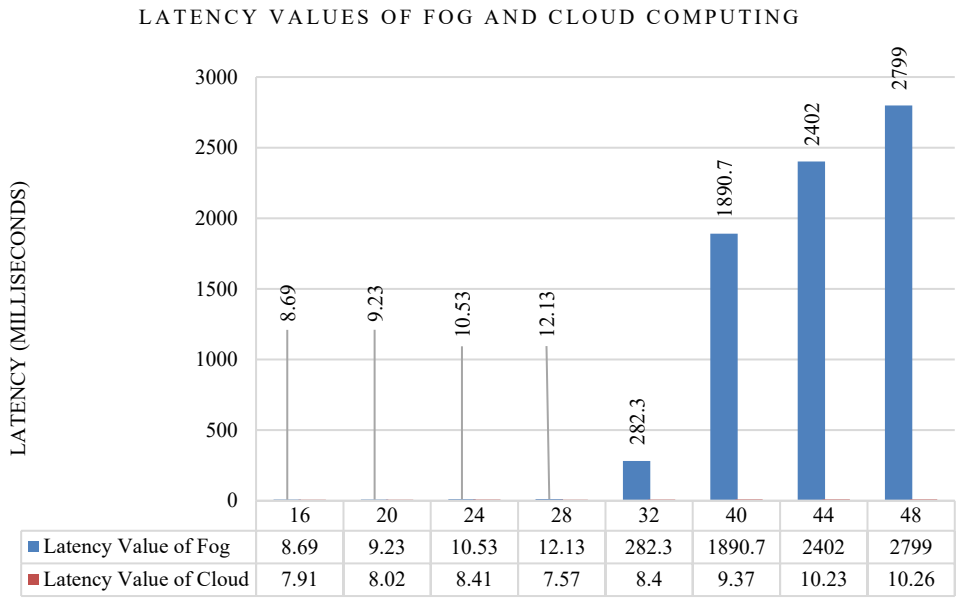


Figure 3. Latency values and comparison of the cloud based and fog computing based results

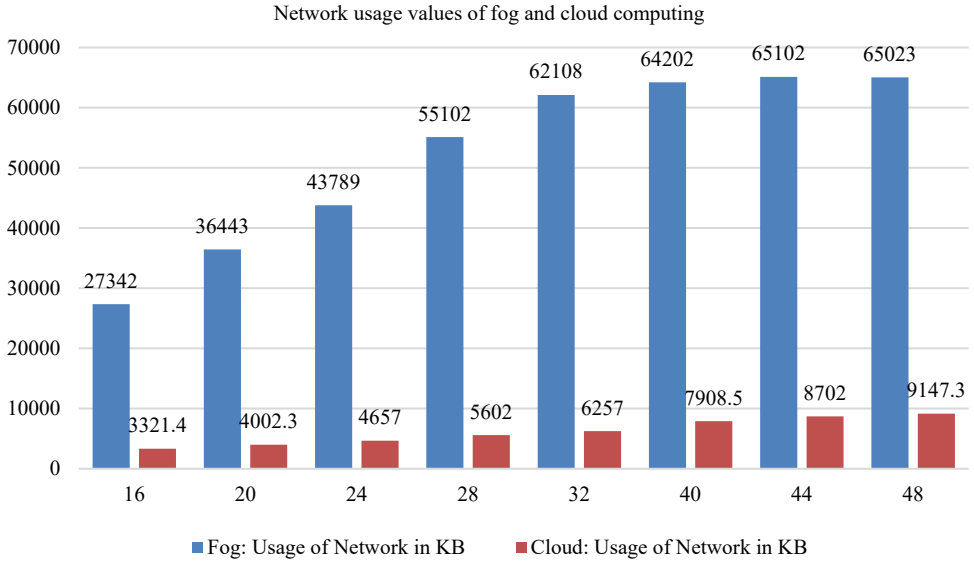
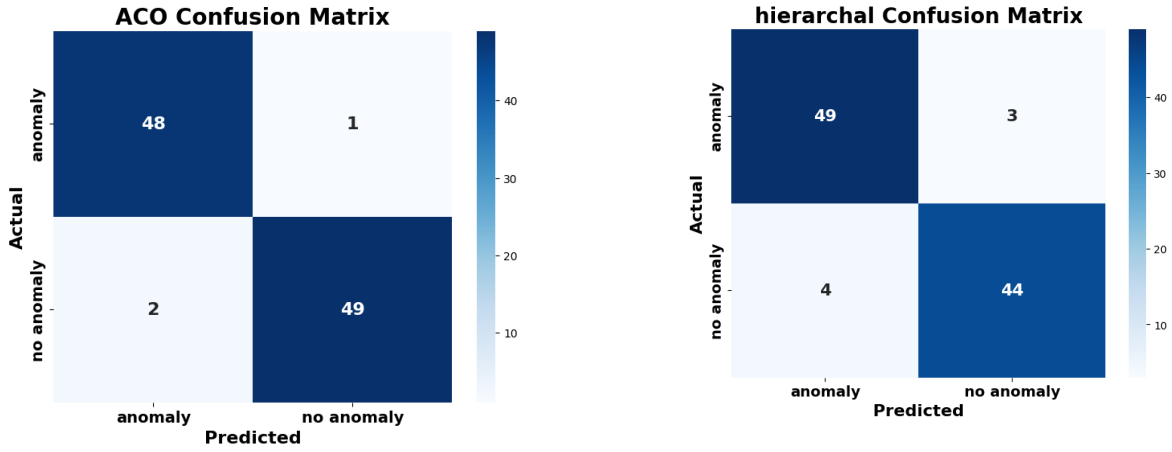


Figure 4. Network use, difference between cloud and fog computing



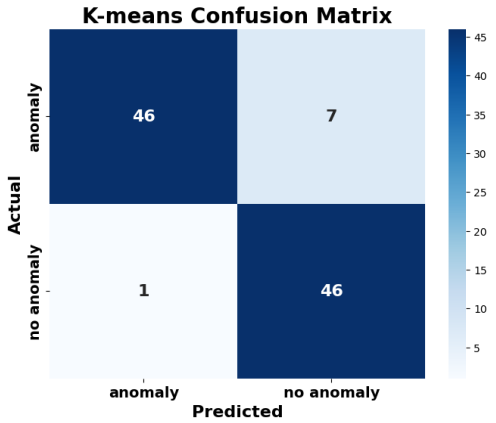


Figure 5. Confusion matrix of three models

According to its confusion matrix, Ant Colony Optimization (ACO) performs exceptionally well in the anomaly detection sector. ACO's capacity to detect anomalies in the dataset is demonstrated by its ability to categorize 49 true positives (*TP*) correctly and without misclassification. It also shows a very low false positive (*FP*) rate of 1, suggesting that it can reduce the number of typical cases that are mistakenly classified as anomalies. Nevertheless, two real positives are mistakenly classified as false positives (*FP*).

ACO emphasizes its appropriateness for anomaly detection activities, where both missed abnormalities and false alarms can have serious repercussions, with this exacting precision and recall. *TP* counts of 46 are shown by K-means clustering, in contrast, with 7 cases being mistakenly labelled as false positives (*FP*). Even with a *TP* count of 44, Hierarchical Optimization exhibits a marginally higher false positive rate (*FP*) of 4 than ACO, with three genuine positives being incorrectly labeled as false positives (*FP*). Despite these differences, ACO is still a strong option for precision-driven anomaly detection tasks due to its outstanding performance, especially in its ability to correctly identify anomalies while reducing false alarms.

Table 2. Performance metrics

Models	*Accuracy*	*Precision*	*Recall*	*F*1-Score
ACO	0.97	0.9702	0.97	0.977
K-means	0.92	0.9266	0.92	0.92
Hierarchal	0.93	0.93011	0.93	0.9299

Three clustering models that are specifically used for anomaly detection are compared in the table: K-means, Hierarchical clustering, and Ant Colony Optimization (ACO). According to ratings of 97%, 0.9702, 0.97, and 0.977 in accuracy, precision, recall, and F1-Score, respectively, ACO is the most successful model. With accuracy rates of 92% and 93%, respectively, and *precision*, *recall*, and *F*1-Score somewhat lower than ACO's, K-means and Hierarchical clustering, while commendable, do not meet ACO's performance measures. ACO's strong performance across all assessment metrics makes it the best option for anomaly detection applications, as demonstrated by these results.

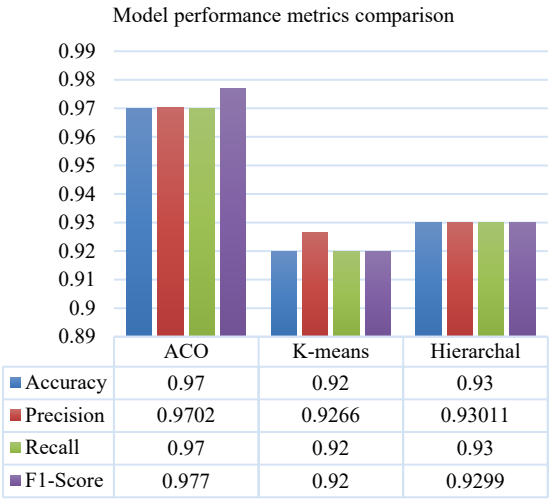
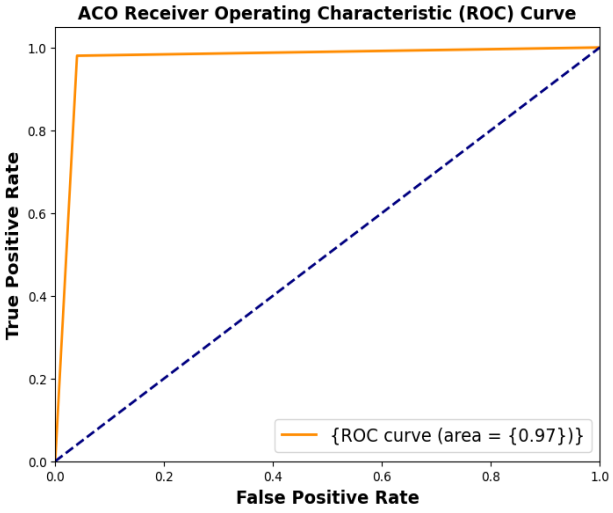
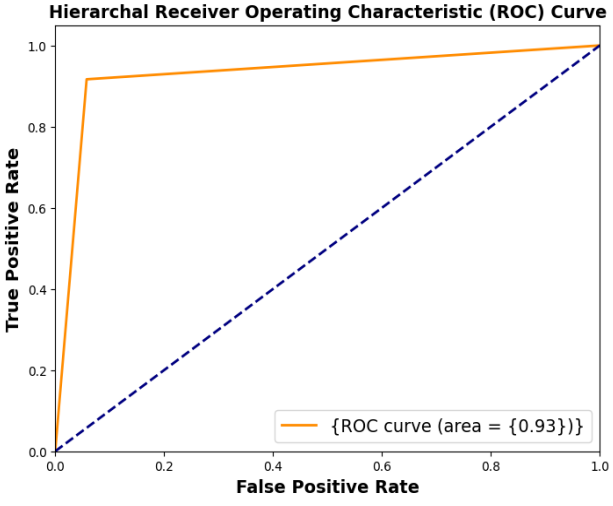


Figure 6. Model performance metrics comparison

The area under the curve (AUC) of 0.97 for the suggested Ant Colony Optimization (ACO) model, compared to 0.92 and 0.93 for K-means and Hierarchical clustering, respectively, highlights the superiority of ACO in anomaly identification. With a greater AUC value, ACO is better able to differentiate between anomalous and normal data items over a range of criteria.





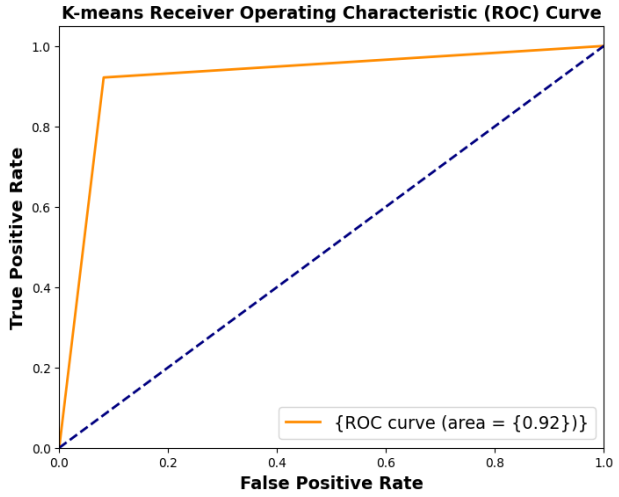


Figure 7. Roc_Curve plots of three models

The capacity of the ACO curve to reach high true positive rates while maintaining low false positive rates is shown by its sharp upward trend, which suggests that it is successful in properly recognizing abnormalities. ACO outperforms K-means and Hierarchical clustering, which shows a somewhat lesser capacity to identify anomalies from normal data, even if they also show good AUC values. ACO is thus the most dependable option for anomaly detection jobs, offering reliable and accurate identification of anomalous data occurrences, according to the ROC curve analysis.

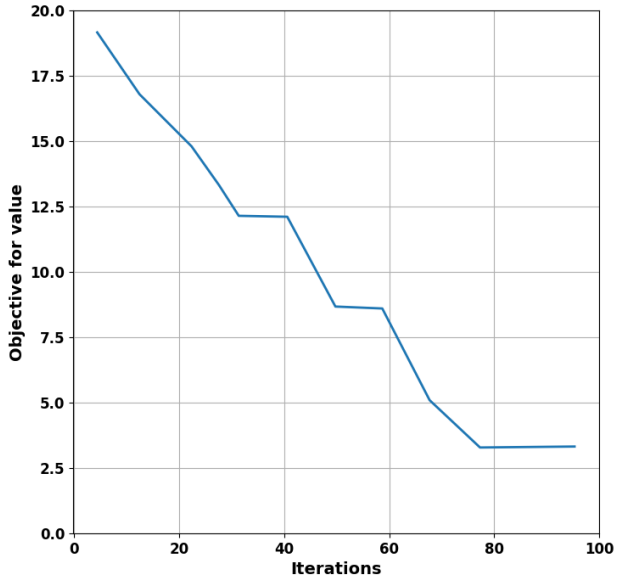


Figure 8. Convergence plot

The value of the objective function is plotted against the number of iterations of the Ant Colony Optimization (ACO) technique of error value filtering in the line graph. Particularly, the objective function value is 20 at iteration 0; it decreases to 17 at iteration 10; it further decreases to 15 at iteration 20; it decreases to 12 at iteration 30; it stays at 12 at iteration 40; it decreases to 8 at iteration 50; it stays at 8 at iteration 60; it decreases to 4 at iteration 80; it decreases to 2 at iteration 90; it stays at 2 at iteration 100; and finally, at iteration 100, it stays at 2. This shows that as the number of iterations rises, the objective function value falls, demonstrating how well the ACO approach reduces error values over time.

7. DISCUSSION

The concept of fog computing and optimization methodologies are discussed together, with an emphasis on how they could be used to boost the reliability and efficiency of intelligent autonomous vehicle parking systems. System optimization has seen an increased reliance on fog computing, which is defined as decentralized processing closer to data sources. Similarly, it is classified as fog computing by this definition that advances the performance of smart parking infrastructures via low latency and improved network efficiency at the network's edge, providing real-time data processing with minimal delays. The merging of advanced technologies is also evident through integrating optimization techniques into fog architectures like those based on Ant Colony

Optimization (ACO) algorithms for anomaly detection. Enhancement of precision and reliability within fog environments can be observed in recall rates and precision achieved by ACO in identifying abnormal occurrences in datasets. Additionally, a line graph depicting the downturn in error values through iterations made by ACO shows how this optimization method relies on iterative processes similar to its underlying principle of iterative data flow approach employed within fog computing. The importance of their input in improving the efficiency and effectiveness of smart car park systems and paving way for more complicated, reliable real-life scenarios is evident because fog computing and optimization techniques ideally complement each other when applied together.

8. CONCLUSION

In conclusion, a combination of fog computing and sophisticated optimization techniques like Ant Colony Optimization (ACO), K-means, and Hierarchical clustering marks a significant step in smart car parking systems. The decentralized processing model of fog computing makes it an important part of smart parking designs due to the latency reduction as well as network efficiency optimization. Comparing fog and cloud architectures is indispensable for understanding the advantages that come with choosing fog computing over cloud computing.

Principally, in real-time data processing, across various sensor configurations considered here, fog latency consistently surpasses cloud latency. Furthermore, the collective upward trend in traffic on networks seen in findings for fog computing indicates more efficient use of network resources compared to those based on the cloud hence enhances its ability to decrease network congestion and overall system performance improvements.

Anomaly detection in terms of accuracy, precision, recall and F1-Score performs better for ACO than K-means and Hierarchical clustering. The accuracy ratings of K-means and Hierarchical clustering are 92% and 93%

respectively while it has a rating of 97%. Its outstanding performance makes ACO the most promising model for anomaly detection tasks in smart car parking systems as it detects anomalies within datasets consistently and without fail. To add on, the convergence graph demonstrates how ACO keeps refining its error values over and over again, just as optimization algorithms naturally get better with time. This is because their iterative nature resonates well with that of data processing in fog computing. In this way, therefore, it makes anomaly detection more accurate and reliable in the context of smart vehicle parking systems within fog computing paradigms.

As this field advances in research and development, the combination of fog computing and advanced optimization techniques is expected to drive further developments, which will make smart car parking systems indispensable for urban mobility maximization among other benefits. In smart cities and urban contexts, we might exploit new designs enabled by the cooperative interplay between fog computing and optimization methodologies to create intelligent, reliable, and efficient parking solutions.

REFERENCES

[1] C. Perera, Y. Qin, J.C. Estrella, S. Reiff Marganiec, A.V. Vasilakos, "Fog Computing for Sustainable Smart Cities: A Survey", ACM Comput. Surv., Vol. 50, No. 3, 2017.

[2] P.H. Cruz Caminha, et al., "SensingBus: Using Bus Lines and Fog Computing for Smart Sensing the City", IEEE Cloud Comput, Vol. 5, No. 5, pp. 58-69, 2018.

[3] S. Jain, S. Gupta, K.K. Sreelakshmi, J.J.P.C. Rodrigues, "Fog Computing in Enabling 5G-Driven Emerging Technologies for Development of Sustainable Smart City Infrastructures", Springer US, Vol. 25, No. 2, 2022.

[4] I. Alrashdi, A. Alqazzaz, R. Alharthi, E. Aloufi, M.A. Zohdy, H. Ming, "FBAD: Fog, Based Attack Detection for IoT Healthcare in Smart Cities", 2019 IEEE 10th Annu Ubiquitous Comput. Electron. Mob. Commun. Conf. Uemcon. 2019, pp. 0515-0522, 2019.

[5] M. Patel, A. Mehta, S. Patel, "Container Migration and Placement in Hybrid Cloud-Fog Environment: Systematic Review", International Journal on Technical and Physical Problems of Engineering (IJTPE), Issue 50, Vol. 14, No. 1, pp. 130-135, March 2022.

[6] K. Gu, X.Y. Dong, W.J. Jia, "Malicious Node Detection Scheme Based on Correlation of Data and Network Topology in Fog Computing-Based Vanets", IEEE Trans. Cloud Comput., Vol. 10, No. 2, pp. 1215-1232, 2022.

[7] L.A. Ajao, S.T. Apeh, "Blockchain Integration with Machine Learning for Securing Fog Computing Vulnerability in Smart City Sustainability", The 1st Int. Conf. Adv. Innov. Smart City, Icaisc 2023, Proc., No. May, 2023.

[8] M. Patel, A. Mehta, N. C. Chauhan, "Design of Smart Dashboard based on IoT Fog Computing for Smart Cities", The 5th Int. Conf. Trends Electron. Informatics, ICOEI 2021, pp. 458-462, 2021.

[9] Y. Sei, A. Oh Suga, "Malicious Node Detection in Mobilewireless Sensor Networks", J. Inf. Process., Vol. 23, No. 4, pp. 476-487, 2015.

[10] A.K. Jha, M. Patel, T. Pawar, "Fog Offloading: Review, Research Opportunity and Challenges", The 2nd Int. Conf. Smart Syst. Inven. Technol. ICSSIT 2019, pp. 1224-1227, March 2019.

[11] K.P. Sinaga, M.S. Yang, "Unsupervised K-Means Clustering Algorithm", IEEE Access, Vol. 8, pp. 80716-80727, 2020.

[12] J. Wang, X. Su, "An Improved K-Means Clustering Algorithm", The 2011 IEEE 3rd International Conference on Communication Software and Networks, pp. 44-46, 2011.

[13] S. Na, L. Xumin, G. Yong, "Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm", The 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, pp. 63-67, 2010.

[14] F. Murtagh, P. Contreras, "Algorithms for Hierarchical Clustering: An Overview", Wiley Interdiscip. Rev. Data Min. Knowl. Discov., Vol. 2, No. 1, pp. 86-97, 2012.

[15] M. Butler, "Hidden Markov Model Clustering of Acoustic Data", Sch. Informatics, Vol. M.Sc., p. 75, 2003.

[16] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, A. Song, "Efficient Agglomerative Hierarchical Clustering", Expert Syst. Appl., Vol. 42, No. 5, pp. 2785-2797, 2015.

[17] C. Blum, "Ant Colony Optimization: Introduction and Recent Trends", Phys. Life Rev., Vol. 2, No. 4, pp. 353-373, 2005.

[18] S.A. Alghamdi, "Novel Trust-Aware Intrusion Detection and Prevention System for 5G Manet, Cloud", Int. J. Inf. Secur., Vol. 21, No. 3, pp. 469-488, 2022.

[19] P. Ebinger, N. Bibmeyer, "Terec: Trust Evaluation and Reputation Exchange for Cooperative Intrusion Detection in MANETs", The 2009 Seventh Annual Communication Networks and Services Research Conference, pp. 378-385, 2009.

[20] R.A. Shaikh, H. Jameel, B.J. d'Auriol, H. Lee, S. Lee, Y.J. Song, "Group-Based Trust Management Scheme for Clustered Wireless Sensor Networks", IEEE Trans. Parallel Distrib. Syst., Vol. 20, No. 011, pp. 1698-1712, 2009.

[21] E. Altulaihan, M.A. Almaiah, A. Aljughaiman, "Cybersecurity Threats, Countermeasures and Mitigation Techniques on the IoT: Future Research Directions", Electron., Vol. 11, No. 20, pp. 1-41, 2022.

[22] M. Abdelhakim, L.E. Lightfoot, J. Ren, T. Li, "Distributed Detection in Mobile Access Wireless Sensor Networks Under Byzantine Attacks", IEEE Trans. Parallel Distrib. Syst., Vol. 25, No. 4, pp. 950-959, 2014.

[23] A. Vempaty, O. Ozdemir, K. Agrawal, H. Chen, P.K. Varshney, "Localization in Wireless Sensor Networks Byzantines and Mitigation Techniques", IEEE Trans Signal Process., Vol. 61, No. 6, pp. 1495-1508, 2013.

[24] K. Gu, X. Ouyang, Y. Wang, "Malicious Vehicle Detection Scheme Based on Spatio-Temporal Features of Traffic Flow Under Cloud-Fog Computing-Based IoVs", IEEE Trans. Intell. Transp. Syst., pp. 1-18, 2024.

[25] M.K. Saroa, R. Aron, "Fog Computing and Its Role in Development of Smart Applications", The 16th IEEE Int. Symp. Parallel Distribute Process with Appl., The 17th IEEE Int. Conf. Ubiquitous Comput. Commun., The 8th IEEE Int. Conf. Big Data Cloud Comput., The 11th IEEE Int. Conf. Soc. Comput. Netw., The 8th IEEE Int. Conf. Sustain. Comput. Commun., No. 7, pp. 1120-1127, December 2018.
[26] A. Iftikhar, K.N. Qureshi, A.A. Altalbe, K. Javeed, "Security Provision by Using Detection and Prevention Methods to Ensure Trust in Edge-Based Smart City Networks", IEEE Access, Vol. 11, No. 7, pp. 137529-137547, November 2023.
[27] R.I. Minu, G. Nagarajan, A. Munshi, K. Venkatachalam, W. Almukadi, M. Abouhawwash, "An Edge Based Attack Detection Model (EBAD) for Increasing the Trustworthiness in IoT Enabled Smart City Environment", IEEE Access, Vol. 10, No. 8, pp. 89499-89508, August 2022.
[28] M.M. Rashid, J. Kamruzzaman, M.M. Hassan, T. Imam, S. Gordon, "Cyberattacks Detection in Iot-Based Smart City Applications Using Machine Learning Techniques", Int. J. Environ. Res. Public Health, Vol. 17, No. 24, pp. 1-21, 2020.
[29] K.S. Awaisi, A. Abbas, M. Zareei, H.A. Khattak, M.U.S. Khan, M. Ali, S. Shah, "Towards a fog Enabled Efficient Car Parking Architecture", IEEE Access, No. 7, pp. 159100-159111, 2019.
[30] M. He, X. Wang, C. Zou, B. Dai, L. Jin, "A Commodity Classification Framework Based on Machine Learning for Analysis of Trade Declaration", Symmetry, Vol. 13, No. 6, p. 964, 2021.
[31] M.P. Patel, A.B. Jha, A.K. Mehta, A.R. Patel, A.J. Nayak, "A Deep Reinforcement Prediction Model for Live VM Migration in Fog", International Journal on Technical and Physical Problems of Engineering (IJTPE), Issue 58, Vol. 16, No. 1, pp. 277-283, March 2024.

BIOGRAPHIES

Name: **Akash**
Middle Name: **Kishorbhai**
Surname: **Mehta**
Birthday: 09.10.1987
Birthplace: Bhavnagar, Gujarat, India
Bachelor: Information Technology, Shantilal Shah Engineering College, Bhavnagar University, Bhavnagar, India, 2009
Master: Information Technology, Shantilal Shah Engineering College, Gujarat Technological University, Gujarat, India, 2014
Doctorate: Student, Computer/IT Engineering, Gujarat Technological University, Ahmedabad, India, Since 2019
Research Interests: Block Chain, Fog Computing, Cloud Computing
Scientific Publications: 7 Papers

Name: **Minal**
Middle Name: **Parimalbhai**
Surname: **Patel**
Birthday: 29.11.1980
Birthplace: Vadodara, Gujarat, India
Bachelor: Computer Engineering, South Gujarat University (Veer Narmad University), Gujarat, India, 2002
Master: Computer Engineering, Sardar Patel University, Gujarat, India, 2007
Doctorate: Computer Engineering, Dharmsinh Desai University, Nadiad, India, 2018
The Last Scientific Position: Assist. Prof, Head of Department, Information Technology Department, Devang Patel Institute of Advance Technology and Research, Charotar University of Science and Technology, Anand, India, Since 2023
Research Interests: Fog Computing, Cloud Computing, Data Science Optimization Techniques
Scientific Publications: 7 Papers, 3 Book Chapters, 3 Theses
Scientific Memberships: ISTE