IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Network Traffic Anomaly Detection Using Machine Learning

[1] Jaya Darshan M, [2]Raja Sankar A, [3]S.Rajeswari, [4]Dr.J.Hemalatha, [5]D.Ramya

[1,2]UG student, Department of Computer Science and Engineering AAA College of Engineering and Technology, Sivakasi.
[3,5]Assistant Professor, Department of Computer Science and Engineering AAA College of Engineering and Technology, Sivakasi.
[4]Professor & Head, Department of Computer Science and Engineering AAA College of Engineering and Technology, Sivakasi.

Abstract: In the connected digital world of today, protecting these networked systems is essential. To ensure the integrity of data transmission over the internet and stop fraud or illegal access, it is essential to identify abnormalities in network behaviour. By spotting malicious attacks in network traffic, intrusion detection systems (IDS) play a critical role in network security. There is no thorough study on anomaly detection utilizing four types of ML models under various network environments, despite the fact that ML techniques have been employed in a variety of fields to address security challenges. The majority of surveys only addressed one form of network (supervised learning).Therefore, we outline in this paper how anomaly detection can be accomplished by unsupervised learning (Support vector machine, k-means clustering technique, NDM) and the current solutions in computer networks, cellular networks, SDN, IoT, and cloud networks. The accuracy of this algorithm is between 94 and 98 percent when compared to supervised learning methods. Additionally, we provide the unsupervised detection approach, which is generally suggested without mentioning the type of network. Experimentally corrected datasets are used to confirm these solutions.

Keywords: Machine learning, anomaly detection, WireShark, unsupervised language, network security, Data Mining, K-means clustering, Computer Network, Support Vector Machine.

1.Introduction:

These days, network traffic classification is dynamic in relation to the hazards posed by evolving technologies. The several classification methods based on machine learning are introduced. By taking into account the variables related to certain application protocols, it assists internet service providers in controlling the network's overall performance. If someone tries to enter the designated traffic lane, it can be used to identify the unfamiliar network. We are able to investigate its features in this way as well. The aforementioned ability to identify unfamiliar networks can also be used to identify the dangers that a network may face as a result of specific security breaches. Quality of Service (QoS) and network security management are other crucial tasks that can be accomplished with effective network classification methods. If we properly categorize our network, we can also block or permit specific network traffic. All things considered, network classification

contributes to the network's overall expansion and effectiveness. The second method to emerge is the payload-based method, which analyzes packets from their respective networks. The main reason this method fails is that it necessitates expensive hardware installations, which are ineffective for encrypted packets. These shortcomings make room for the machine learning technique, which is currently in use because of its effectiveness in producing results and resemblance to real-world data. This method involves turning labelled classes into models, training and testing them, and then using accuracy to verify that the models are right. The following is an explanation of the paper's contributions. After discussing the various approaches, we use machine learning techniques to a network data set and compare several algorithms to determine which is most appropriate for network traffic analysis. Using the Wireshark tool, we gather the KDDCUP99 dataset's features.

2.Literature Survey:

[1] Oviya G, Preethi J, Thoufeeq A, "Network Traffic Anomaly Detection Using ML", International Journal of Progressive Research in Engineering Management and Science (IJPREMS), Volume 4, Issue 4, 2024. Oviya proposed Through linked infrastructures, networked computer systems have become essential to almost every facet of contemporary life, facilitating social, commercial, and governmental endeavors. As the Internet grew quickly, these systems were more frequently the focus of cyberattacks, endangering their availability, secrecy, and integrity. Researchers used machine learning techniques to identify significant trends in massive datasets for enhanced security in order to combat such invasions. Algorithms for machine learning were created with the ability to handle previously encountered examples and generalize from existing data. In order to improve intrusion detection and system resilience, the suggested approach highlighted the significance of sound generalization.

[2] Manoharan Premkumar et al., "Augmented Weighted K-means Grey Wolf Optimizer: An Enhanced Metaheuristic Algorithm for Data Clustering Problems", Scientific Reports, Volume 14, Article 5434, 2024. This paper focus on Network anomaly detection has grown more difficult as a result of the irregular traffic and the quick development of network technologies. Unsupervised techniques have poor accuracy, and existing supervised techniques were unable to identify unknown attacks. Researchers used a unique technique, DPC-GS-MND, based on grid screening and mutual neighborhood degree, to propose a clustering-based anomaly detection model in order to address issue. With automatic cluster center selection, the suggested approach enhanced clustering accuracy while lowering computational cost. DPC-GS-MND demonstrated tremendous potential for complex network settings by achieving higher accuracy and efficiency in experiments conducted on the KDDCup99 and CIC-IDS-2017 datasets.

[3] Song Wang, "Machine Learning for Network Defense," IEEE preprint, vol. arXiv:2306.13017, June 2023. This paper focus on the technology advances and the number of connected devices increases quickly, anomaly detection in networks continues to be a concern. With the increasing diversity of cyberattacks, traditional approaches frequently prove inadequate. A versatile and efficient substitute for identifying intrusions in a variety of network types is machine learning. Its benefits, methods, and implementation have all been studied. The capabilities of several machine learning models in anomaly detection are highlighted via comparative assessments.

[4] Zhang, H. "A generalized K-means algorithm", In Proceedings 2000 International Conference on Artificial Intelligence, (IC-AI'2000) (pp. 43–49), 2023 Zhang H proposed the K-means is the simplest algorithm for grouping analysis, which was considered a fundamental data mining tool. However, problems with classical K-means included limited applicability of SSE and unstable K value selection. Researchers suggested an enhanced K-means approach based on clustering reliability analysis to get around these problems. The issues of unequal density and fluctuating data volumes were successfully resolved by the suggested approach. Consequently, it was able to attain more accurate and consistent clustering performance.

[5] Lavanya Gundeboina et al., "Network Traffic Analysis Using Machine Learning", International Journal of Emerging Technologies and Innovative Research, Volume 13, Issue 4, 2023. This paper focus

on the large volumes of data were transferred across networks as a result of the sharp rise in network traffic, leaving them open to intrusions and attacks. Researchers concentrated on security measures to find possible risks and discover anomalies in order to resolve this. To give a thorough overview of the most recent developments in anomaly detection throughout the last five years, a study was carried out. The study's conclusion focused on issues that still needed to be fixed in order to improve anomaly detection systems.

[6] Kavitha D, Ramalakshmi R, "Machine learning-based DDoS attack detection and mitigation in SDNs for IoT environments," Journal of the Franklin Institute, vol. 361, p. 107197, 2023. Kavitha and Ramalakshmi's work explores the use of machine learning algorithms for detecting and mitigating DDoS attacks in SDN-enabled IoT networks. They analyze the challenges posed by large-scale IoT deployments, where traditional DDoS mitigation techniques often fall short. The authors propose a hybrid model that combines supervised and unsupervised learning algorithms, such as Support Vector Machines (SVM) and clustering, to enhance the accuracy of attack detection. Their model significantly reduces false positives and improves the efficiency of mitigation strategies.

[7] Purnendra Kumar et al., "Network Traffic Analysis and Prediction Using Machine Learning", International Journal of Research Publication and Reviews, Volume 4, Issue 8, pp. 2071–2075, 2023. It has been acknowledged that improving network security and performance requires network traffic analysis. In response to the growing network traffic and developments in artificial intelligence, the article examined several machine learning techniques used for traffic analysis. The methods used in network traffic analysis were emphasized in the review. The suggested techniques showed how well ML works to address important network security problems.

[8] Linangchen Chen et al., "Network Anomaly Detection Using Mutual Neighbour Degree", Journal of Intelligent & Fuzzy Systems, Volume 43, Issue 3, 2022. Linangchen Chen proposed a one popular unsupervised machine learning technique for locating discrete, non-overlapping clusters based on minimum squared distance is K-means. Finding the best starting centroids for the first iteration was a significant algorithmic problem. In order to minimize iterations and execution time, researchers suggested an effective technique for choosing initial centroids. Real-world datasets, such as COVID-19 and patient data, as well as a synthetic dataset with 10 million instances with time and iteration count were used to assess the suggested approach.

[9] Md. Zubair et al., "An Improved K-means Clustering Algorithm Towards an Efficient Data-Driven Modeling", Annals of Data Science, 2022 (Online First). This paper focus on Network anomaly detection has grown more difficult as a result of the irregular traffic and the quick development of network technologies. Unsupervised techniques have poor accuracy, and existing supervised techniques were unable to identify unknown attacks. Researchers used a unique technique, DPC-GS-MND, based on grid screening and mutual neighborhood degree, to propose a clustering-based anomaly detection model in order to address issue. With automatic cluster center selection, the suggested approach enhanced clustering accuracy while lowering computational cost. DPC-GS-MND demonstrated tremendous potential for complex network settings by achieving higher accuracy and efficiency in experiments conducted on the KDDCup99 and CIC-IDS-2017 datasets.

[10] Mohammed Hussein Thwaini, "Anomaly Detection in Network Traffic Using Machine Learning for Early Threat Detection", Data & Metadata, Volume 1, Article 34, 2022. This paper focus on the network usage increased quickly, a large amount of data was exchanged, increasing the vulnerability of networks to breaches and assaults. Researchers underlined the necessity of security measures that can recognize threats and detect anomalies in order to handle this. A noninvasive evaluation of recent developments in anomaly detection systems was carried out. The assessment emphasized current issues and potential directions for improving anomaly detection systems in the future.

[11] Azhar Rufai, "Clustering Algorithms: K-Means and Variants", International Journal of Advanced Computer Science and Applications (IJACSA), Volume 11, No. 12, 2020. Azhar Rufai proposed the K-means, a frequently used technique based on randomly chosen initial centroids, was one of the clustering strategies suggested to be used to group comparable data items together. However, the algorithm's effectiveness was frequently impacted by this random selection. In order to overcome this, scientists suggested a modified K-means method that determined initial centroids in a deterministic manner. The number of iterations needed was decreased by the suggested approach. Consequently, it reduced the total elapsed time, which enhanced clustering performance.

[12] Ertoz, L., Lazarevic, A., Kumar, V., Tan, P. N., & Srivastava J., Data Mining for Intrusion Detection", In Next Generation Data Mining (MIT Press), vol.Book Chapter, 2004. This chapter explores various clustering and anomaly detection algorithms tailored for network intrusion detection systems (NIDS). It focuses on the challenges of applying unsupervised learning techniques, such as scalability and the presence of noisy data. The authors present comparative evaluations of clustering-based approaches, highlighting their effectiveness in identifying novel attacks with minimal false positives. The study supports hybrid models combining misuse detection with anomaly detection to improve accuracy.

[13] Paul Dokas et al., "Data Mining for Network Intrusion Detection", University of Minnesota Technical Report, Department of Computer Science, 2002. This paper reviews literature in the area of accident reporting and knowledge-based systems. Focuses on incident learning, with the goal of developing systems that support decision-making based on past incident data. Discusses the classification of incident types, root cause analysis, and use of ontologies to formalize incident knowledge.Works on incident databases and knowledge management systems. Approaches to semantic analysis and formal representation of incidents.

[14] Esposito, F., Malerba, D., & Lisi, F. A., "Intrusion Detection in Computer Networks by Pattern Recognition", Pattern Recognition and Image Analysis, Approx. early 2000s. This research applies pattern recognition methods, particularly decision tree induction and rule-based classifiers, to intrusion detection. The authors detail the importance of feature selection and the transformation of raw network data into high-level behavioral patterns. Their experiments show that data-driven approaches can effectively recognize both known and novel intrusions, offering interpretability and adaptability across different network configurations .

[15] Susan M. Bridges and Rayford B. Vaughn, "Intrusion Detection via Fuzzy Data Mining", Twelfth Annual Canadian Information Technology Security Symposium, Ottawa Congress Centre, 2000. The paper positions itself within the domain of incident detection and management (IDM), particularly in the context of social media and crowd-sourced data. Cited works include methods of event detection, topic modeling, and natural language processing for identifying incidents. There's a specific interest in real-time systems and the integration of multiple data streams (e.g., social media, sensor networks).

[16] Lee, W., Stolfo, S. J., & Mok, K. W., "ADAM: Detecting Intrusions by Data Mining", IEEE Workshop on Information Assurance and Security, 2000

The ADAM project introduces a scalable and efficient system for detecting intrusions using data mining techniques. The system utilizes audit data to learn patterns that indicate abnormal or malicious activity. It emphasizes the importance of frequent pattern mining and classification methods in real-time environments. ADAM's architecture involves preprocessing audit data, applying classification algorithms, and generating real-time alerts, making it highly adaptable to dynamic network environments

3.Research Methodology:

K-means clustering is used to combine similar data points into clusters once network traffic data has been collected and transformed into feature vectors. Any network behavior that substantially deviates from the established "normal" patterns inside the clusters is essentially indicated by flagging data points that fall

significantly from the cluster centroids as possible anomalies. Using the K-means clustering algorithm, this is a recommended technique for detecting network anomalies.

3.1 Network Data Mining (NDM)

The practice of deriving significant patterns, connections, and insights from data produced within computer networks is known as Network Data Mining (NDM). In order to find hidden information in network traffic, communication logs, or social network structures, it integrates methods from data mining, machine learning, and network analysis. Understanding how networked systems behave, spotting irregularities like fraud or intrusions, and improving performance all depend on NDM. By spotting odd patterns in data flows or access attempts, NDM makes it possible to detect suspicious activity. By examining connection patterns in social networks, it can uncover hidden communities or powerful users. Modern networks are complicated, requiring sophisticated algorithms that can handle vast amounts of dynamic, frequently unstructured data. As a result, NDM is a rapidly developing topic that is crucial to both industry and academics.

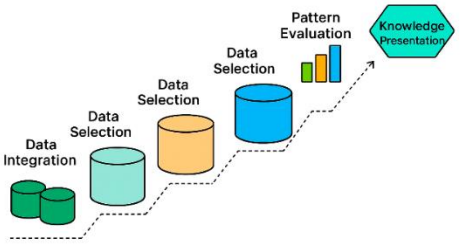


Fig 1. Knowledge Discovery in Database

3.2 The Dataset Model

NDM is applied to vast volumes of monitoring data describing packet, flow, or connection properties and statistics. To extract the appropriate rules and patterns, this data are often processed and stored in a database shown in **Fig 1**. Five distinct processing phases can be identified using the KDD (Knowledge Discovery in Databases) model.

- **Description of Data Integration:** Information is gathered and aggregated from various sources, including logs, databases,andsensors.
  The goal is to create a cohesive picture of the data so that it can be analyzed consistently across various systems.
- **Description of Data Cleaning:** Missing values, errors, and duplication are found and either eliminated or repaired.
  The goal is to guarantee proper analysis and enhance data quality.
- **Description of Data Selection:** Based on the analysis's goals, pertinent data is chosen.
  The goal is to minimize the amount of data and concentrate solely on the characteristics required for mining.
- **Data transformation** is the process of transforming data into suitable formats or structures, such as aggregation or normalization.
  The goal is to improve interpretability and standardize data so that it may be used for mining.
- **Data mining** is the process of using algorithms to find trends, patterns, or connections.
  The goal is to get significant discoveries like clusters, connections, or classifications.
- **Description of Pattern Evaluation:** The validity, originality, and utility of the mined patterns are assessed.
  The goal is to eliminate redundant or unnecessary results so that only important patterns remain.
- **Description of the Knowledge Presentation**: The verified patterns are presented or displayed in a way that is easy for human to understand.
  The goal is to assist in decision-making by effectively communicating the findings to stakeholders.

3.3 Dataset Fragmentation

The three parts that make up the KDD Cup 99 intrusion detection system lists in Table 1. The "10% KDD" dataset contains just 22 different attack methods, the majority of which fall within the denial of service category. The "Corrected KDD" dataset has different statistical distributions than the "10% KDD" or "Whole KDD" datasets. There are 37 different kinds of attacks in it. The number of recordings in each attack category is shown in Table 1

Table-1 The Intrusion Detection Dataset in term of numbers

Dataset	DoS	Probe	U2R	R2L	Normal
10% KDD	391457	4107	55	1126	97277
Corrected_KDD	229853	4166	70	16344	60599
Whole_KDD	3883380	41102	58	1126	972780

Domain-specific characteristics include the number of files created, the number of unsuccessful login attempts, the protocol type, the time, the number of bytes exchanged, and whether or not a root shell was acquired. The suggested experiment makes use of the Corrected KDD. Table 2 lists the four categories (R2L, Dos, U2R and Probe) into which the dataset's 37 attack types as shown in Fig 2

Table-2 Attack types with their categories

Category	Types of Attack
R2L	Httptunnel, ftp_write, worm, imap, xlock, multihop, warezmaster, named, snmpguess, phf, , snmpgetattack, xsnoop, guess_password ,sendmail
DoS	udpstorm apache, mailbomb, back, neptune ,land, smurf, teardrop, processtable , pod
U2R	xterm ,buffer_overflow, rootkit, ps, loadmodule, perl, sqlattack
Probe	Satan, nmap, portsweep, mscan, ipsweep, saint

Table-3 Classification of NDM Approaches.

Approach	Dataset	Features	Algorithm	Knowledge
L. Ertoz et al	Packets (tcpdump) In TCP connections & UDP connections	Time based features	Outlier Detection	Anomalies in analysed data
P. Dokas et al	Flow Records (Cisco netflow, ISMP)	Time based and connection based features	Outlier Detection	Anomalies in analysed data
Esposito et al	Packets (tcpdump)	Connection records	Rule learning	Classification rules
C. S. Jajodia et al	Packets (tcpdump)	Connection records	Rule learning	classification rules
Luo et al	Packets (tcpdump)	Counters for TCP SYN/FIN/RST packets and number of different destination ports	Rule learning	Fuzzy association &classification rules

Our approach	Flow records (computer network, IPFIX)	Counters of bytes, packets, active flows for different time intervals and service-specific ports	K-Means Clustering	Centroids for normal and anomalous clusters

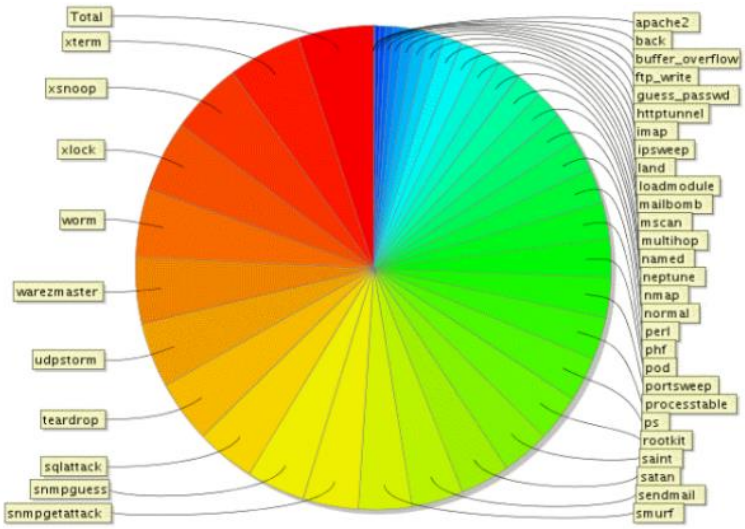


Fig 2 The 37 types of attack found in corrected KDD

Based on related studies that used rule-learning algorithms on KDD-CUP'99 competition submissions of connection data with similar properties. Anomalies are detected using outlier identification using time-based and connection-based feature sets. The local outlier factor (LOF) detection method was chosen to be integrated into the Minnesota Intrusion Detection System (MINDS). In MINDS, connection records were classified as highly uncommon by the LOF algorithm.

Esposito et al. and Luo et al. use Tool-Diag, a pattern recognition toolbox, to identify a tiny subset of the connection attributes used by Dakos et al. that has the maximum discriminating power. The selected features are then subjected to a rule-learning algorithm to generate network behavior patterns. Unfortunately, the connection features that the authors eventually used are not disclosed. Barbara et al. used classification and association techniques on connection logs and achieved very good results in the DARPA'99 intrusion detection test. The usage of fuzzy frequent episodes and fuzzy association rules is recommended.

While MINDS clusters typical flow data into a single cluster, we believe that normal and anomalous traffic form separate clusters in the feature space. We employ K-means clustering to find groupings for both normal and abnormal data. Finally, we use a fast distance-based method to classify new monitoring data and detect outliers.

4. Data and their Extracted Features

First, the transport protocol and specified port numbers typical for frequently used services are utilized to categorize flow records. For instance, TCP & UDP records that include port 80 as their source or destination are categorized as HTTPS/HTTP traffic or web traffic. This classification was made because, depending on the service or application, regular traffic can seem extremely different. Thus, it is possible to use the K-means clustering method independently for distinct services defined by their (protocol, port) pairs by differentiating flows based on their protocol and service-specific port numbers. For TCP, UDP,SMTP or ICMP traffic, flow records that do not belong to any of the specified service classes are allocated to the appropriate default class.

Each dataset contains the following features:

- Total number of bytes sent from and to the specified port during the time period under consideration.
- Total number of packets sent from and to the specified port during the time period under consideration.
- The number of distinct source-destination pairs 1 that are seen throughout the time period under consideration and match the specified service-specific port and protocol

This feature selection was motivated by the fact that the number of packets and bytes enables the detection of abnormalities in traffic volume, and these third characteristic aids in the detection of distributed attacks, network and port scans, and other activities that lead to a rise in source-destination pairs.

3.1 K-Means cluster method

The clustering analysis procedure known as K-means clustering divides objects into K distinct groupings according to the values of their features. The feature values of objects that belong to the same cluster are comparable. The number of clusters, K, is a positive integer that must be provided beforehand. The K-means clustering algorithm consists of the following five steps:

1) Establish K as the number of clusters.
2) Set up the centroids of the K cluster. By grouping all objects into K clusters at random, calculating each cluster's centroids, and confirming that each centroidal is distinct from the others, this can be accomplished. An alternative is to initialize the centroids to K randomly selected, distinct objects.
3) Calculate the distances to each cluster's centroids by iterating over every object. Each object should be assigned to the cluster that has the closest centroid.
4) Recalculate both changed clusters' centroids.
5) Continue with step 3 until the centroids stop changing.

A distance function is required in order to compute the distance (i.e. similarity) between two objects. The most commonly used distance function is the Euclidean one which is defined as:

$$d(x, y) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2}$$

However, calculating and inverting the covariance matrix is computationally demanding for feature vectors with a large number of dimensions.

For initial evaluation of the proposed anomaly detection approach, we used a weighted Euclidean distance defined as:

$$d(x,y) = \sqrt{\sum_{i=1}^{m}\left(\frac{x_i - y_i}{s_i}\right)^2}$$

where S_i is an empirical normalization and weighing factor of the i^{-th} feature. The larger S_i and the smaller is the influence of the i^{-th} feature on the distance. We found that good coefficients for the number of packets, bytes, and source-destination pairs (src-dst) are $S_{packets} = S_{bytes} = 5$ and $S_{src-dst} = 1$.

We use training datasets that may contain both regular and abnormal traffic without being explicitly identified as such before applying the K-means clustering technique. This method's justification is based on the idea that typical and unusual traffic create distinct feature space clusters. Naturally, there may be outliers in the data that do not fit into a larger cluster, but as long as there are few outliers, the K-means clustering procedure is unaffected. As previously stated, the preconfigured services are clustered separately based on their usual (protocol, port) combination, as are the default classes that encompass the other flows that are characterized just by the protocol value.

The training data is divided into K clusters by the clustering method, but it is not possible to tell whether a cluster represents periods of typical or unusual activity. Either heuristics or manual decision-making are required. For instance, an unusual cluster may be indicated by a greater average number of packets. Clusters may be found in close proximity to one another. This may be due to a number of factors: Either the

training data is extremely homogeneous, for example, because it contains no anomalous traffic or because the aberrant traffic appears to be very similar to regular traffic, or the number of clusters K has been poorly chosen.

4.2. K-Means Classification and outlier Detection

Cluster Classification: The weighted Euclidean distance function is used to determine the distances to the cluster centroids of the respective traffic class. If an object is nearer the normal cluster centroid than the anomalous one, it is considered normal, and vice versa. Fig 3 uses a two-dimensional feature space to demonstrate this: Since object P is nearer the normal cluster, it is considered normal. This distance-based categorization makes it possible to identify known types of anomalies, such as unusual traffic that shares traits with the training datasets.

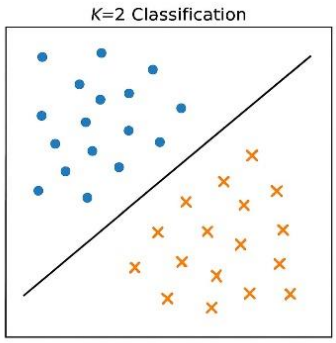


Fig 3 Classification for K=2

Identifying outliers: An object is considered an outlier if it dramatically deviates from the majority of other objects. As a result, it can be regarded as unusual. Only the distance to the proper centroid of the normal cluster is computed for outlier detection. An object is considered an outlier and anomaly if its distance from the centroid is greater than a predetermined threshold, Fig 4 illustrates this, showing that P2 and P3 are outside the circle. Outlier detection may be less accurate in identifying recognized types of anomalies because it does not utilize the anomalous cluster centroid like the classification approach does. However, it makes it possible to find novel anomalies that aren't present in the training datasets.

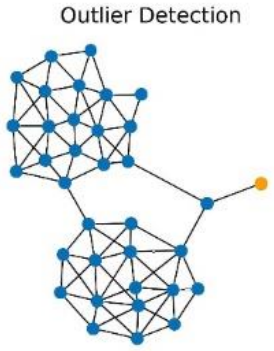


Fig 4 Outlier Detection

5.Experimental work

Our experimental work has tested and evaluated using a labeled or unlabeled dataset of network traffic records, typically from sources such as the KDD Cup dataset, which is a perfect candidate for the application of clustering and anomaly detection techniques because it captures a wide range of simulated network activity, both malicious and normal. Preprocessing involved extracting relevant features such as packet sizes, IP addresses, handle the missing values through imputation or deletion removing duplicates. These steps ensured the quality and relevance of the data for modelling training.

5.1 Related Work

Even though ML approaches have been applied in a number of sectors to solve security challenges, this study examines anomaly detection in unsupervised machine language using four different types of ML models under different network conditions. Only one network type (supervised learning) was covered in most surveys. Thus, we describe the existing solutions in cloud, IoT, SDN, cellular, and computer networks in this study, along with the use of unsupervised learning for anomaly detection. We also present the unsupervised detection method, which is commonly recommended without specifying the network type. These solutions are validated using experimentally corrected datasets.

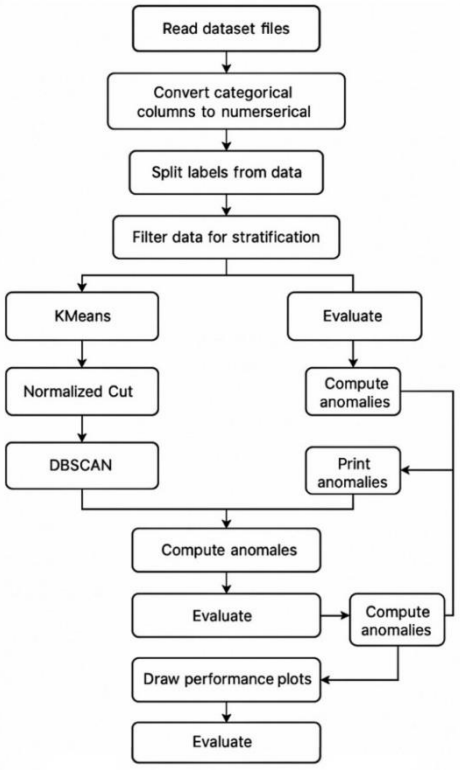
5.2. Data Acquisition and Preprocessing

Using the gdown package, the code first downloads the KDD Cup 1999 dataset. This dataset, which includes a range of simulated network intrusions in a military network context, serves as a typical benchmark for intrusion detection systems. There are various versions of the dataset available:

- *kddcup.data.gz*: The full dataset.
- *kddcup.data_10_percent.gz*: A 10% subset of the full dataset.
- *corrected.gz*: Test data with corrected labels.

The code downloads all three versions and unzips them into a designated directory /Datasets/

5.3 Code Demonstration Flow



5.4 Data Loading and Transformation

The downloaded datasets are loaded into Pandas DataFrames using the ***read_csv*** function. These DataFrames provide a structured way to work with the data. Since the dataset contains categorical features (e.g., protocol type, service, flag), a custom function ***convert_categorical_columns*** is defined to transform these features into numerical representations. This transformation is crucial because many machine learning algorithms, including clustering algorithms like K-Means, require numerical input.

The ***convert_categorical_columns*** function works by iterating through the columns of the DataFrame and identifying categorical columns based on their data type (object). For each categorical column, it creates a dictionary to map unique categorical values to numerical values. If a dictionary for a particular column already exists (passed as an argument), it reuses that dictionary to ensure consistency across different DataFrames. The function then replaces the categorical values in the column with their corresponding numerical values.

After the transformation, the DataFrames are converted into NumPy arrays using the ***to_numpy*** function. NumPy arrays are efficient data structures for numerical computations. The data is then split into features (data) and labels (labels).

Finally, to free up memory and potentially improve performance, the original DataFrames and intermediate variables.

5.5. Clustering with K-Means

5.5.1 K-Means Implementation

A custom K-Means class is implemented from scratch. This implementation provides flexibility and allows for understanding the underlying algorithm's logic. The class has methods for:

- ***_init__*** : Initializes the object.
- ***__initialize_centroids*** : Randomly selects initial centroids from the training data, ensuring they are distinct.
- ***__compute_clusters_indices*** : Calculates the distance between each data point and the centroids and assigns the data point to the cluster with the nearest centroid.
- ***__assign_clusters*** : Assigns data points to clusters based on their calculated distances to the centroids.
- ***__update_centroids*** : Recalculates the centroids of each cluster by taking the mean of all data points assigned to that cluster.
- ***__kMeans***: Performs the main K-Means iteration, assigning clusters and updating centroids until convergence or a maximum number of iterations is reached.
- ***__print_clusters_info***: Prints information about the clusters, including their sizes.
- ***__compute_sse***: Calculates the sum of squared errors (SSE), a measure of the within-cluster variance.
- ***fit***: Fits the K-Means model to the training data, performing multiple restarts with random initializations to find a better solution.
- ***predict***: Predicts the cluster assignments for new data points based on the learned centroids.
- ***get_centroids***: Returns the learned centroids.

5.5.2 Applying K-Means

The K-Means algorithm is applied to the reduced dataset ***(reduced_data)*** using two different values for ***k*** (the number of clusters): ***7 and 15***. These values are stored in the ***k_values*** list. A dictionary ***kmeans_dict*** is used to store the K-Means objects for each value of k

For each value of ***k***, a K-Means object is created and fitted to the reduced data using the fit method. The ***n_iterations*** parameter specifies the maximum number of iterations allowed for the algorithm to converge.

5.6 Normalized Cut

5.6.1 Normalized Cut Implementation

A function named normalized_cut is used to implement the Normalized Cut algorithm. The goal of this graph-based clustering technique is to divide the data points into clusters so that there is a high degree of similarity between the data points within the clusters and a low degree of similarity between the data points in different clusters. It employs the spectral clustering technique, which entails figuring out the similarity graph's eigenvectors and Laplacian matrix. The Normalized Cut algorithm's primary steps are:

- **Constructing the Similarity Matrix**: A similarity matrix is built using the Radial Basis Function (RBF) kernel, where the similarity between two data points is calculated based on their Euclidean distance.
- **Computing the Laplacian Matrix**: The Laplacian matrix is computed as the difference between the degree matrix and the similarity matrix. The degree matrix is a diagonal matrix where each element represents the sum of the similarities of a data point to all other data points.
- **Eigendecomposition**: The Laplacian matrix is eigendecomposed to obtain its eigenvalues and eigenvectors.
- **Selecting Eigenvectors**: The eigenvectors corresponding to the smallest eigenvalues are selected to form a new representation of the data.
- **Clustering:** K-Means is applied to the new representation of the data to obtain the final clusters.

5.6.2 Applying Normalized Cut

A tiny portion of the data (training_data), approximately 0.15% of the whole dataset, is subjected to the normalized_cut function. The algorithm's computational complexity is decreased by using this subset. The required number of clusters is indicated by setting the k parameter to 11. The width of the Gaussian kernel used to calculate the similarity between data points is controlled by the g parameter, which stands for the gamma value for the RBF kernel. The normalized_cut function's output consists of:

- **clusters**: The cluster assignments for each data point in the training data.
- **new_training_data**: The new representation of the training data obtained by selecting the eigenvectors of the Laplacian matrix.
- **centroids**: The centroids of the clusters obtained by applying K-Means to the new representation of the data.

5.7. DBSCAN

5.7.1 DBSCAN Implementation

The sklearn.neighbors module's BallTree class is used to build the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm. A density-based clustering technique called DBSCAN can find clusters of any size or form. Clusters are defined as dense regions of data points that are divided by less dense regions. The DBSCAN algorithm's primary steps are:

- **Finding Core Points**: A data point is considered a core point if it has at least ***min_samples*** data points within a distance of ***eps*** (epsilon) from it.
- **Creating Clusters**: Core points that are within a distance of ***eps*** from each other are assigned to the same cluster.
- **Assigning Border Points**: Data points that are within a distance of ***eps*** from a core point but do not have enough neighbors to be considered core points themselves are assigned to the cluster of the core point.

- **Identifying Noise Points**: Data points that are neither core points nor border points are considered noise points and are not assigned to any cluster.

The ***create_cluster*** function is a helper function that recursively assigns data points to a cluster based on their density connectivity. It starts with a core point (root) and expands the cluster by adding neighboring points that are within a distance of ***eps*** and have at least ***min_samples neighbors.***

- ***BallTree*** for efficient radius-based neighbor queries.
- Label core, border, and noise points based on density (***eps, min_samples***).
- Expand clusters from core points.

5.7.2 Applying DBSCAN

The training data (training_data) is subjected to the DBSCAN algorithm with min_samples set to 41 and eps set to 10. These variables regulate the clusters' density and the bare minimum of points needed to create a cluster. An array clusters with the cluster assignments for every data point in the training data is the result of the DBSCAN function. Cluster -1 data points are regarded as noise points.

5.8. Evaluation Metrics and Anomaly Detection

The remaining sections of the code focus on evaluating the performance of the clustering algorithms and identifying anomalies in the data.

5.8.1 Evaluation Functions

Several functions are defined to calculate evaluation metrics for the clustering results. These metrics include precision, recall, F1 score, conditional entropy, and purity. These metrics help assess the quality of the clusters produced by each algorithm.

- **Purity**: Purity measures how "pure" each cluster is i.e., how many data points in a cluster belong to the majority true class label. In this, it will measures the proportion of data points in a cluster that belong to the most frequent class in that cluster.

$$\textbf{Purity} = \frac{1}{n}\sum_{c \in C} \boldsymbol{max}_{y \in Y} |\boldsymbol{c} \cap \boldsymbol{y}|$$

- **Recall:** Recall measures how well the model retrieves all relevant instances of a particular class. In clustering, it's often adapted to compare clusters to ground-truth labels. In this, it will measures the proportion of data points belonging to a particular cluster that are correctly assigned to that cluster.

$$\textbf{Recall} = \frac{\textbf{True Positives}}{\textbf{True Positives+False Negatives}}$$

- **F1 Score:** F1 Score is the harmonic mean of precision and recall. It balances the two, especially useful when classes are imbalanced. In this, it will measures the harmonic mean of precision and recall, providing a balanced measure of performance.

$$\textbf{F1 Score= 2 x } \frac{\textbf{Precision X Recall}}{\textbf{Precision+Recall}}$$

- **Conditional Entropy:** Conditional entropy quantifies the uncertainty of ground-truth class labels given the cluster assignments. In this it will measures the uncertainty in the cluster assignments given the true labels of the data points.

$$\textbf{H(Y|C)= } -\sum_{c \in C}\sum_{y \in Y} \boldsymbol{P(c,y) log} \frac{\boldsymbol{P(y|c)}}{\boldsymbol{P(y)}})$$

5.8.2 Applying Evaluation Metrics

The evaluation metrics are applied to the clustering results obtained by K-Means, Normalized Cut, and DBSCAN. The results are stored in lists and then visualized using plots to show the relationship between the metrics and the number of clusters (***k***).

5.8.3 Anomaly Detection

Anomalies (outliers) are identified in the data based on the clustering results. Anomalies are data points that do not belong to any cluster or are assigned to a cluster with a different label than the majority of points in that cluster. A function ***compute_anomalies*** is defined to identify these anomalies.

The ***compute_anomalies*** function iterates through the clusters and identifies data points that do not belong to the dominant class in their cluster. These data points are considered anomalies.

The anomaly detection function is applied to the clustering results of DBSCAN and Normalized Cut. The identified anomalies are printed to the console.

6. Conclusion

In this study, we investigated the application of the K-Means clustering algorithm to the detection of anomalies in network data. Real-time and effective anomaly detection has emerged as a crucial component of network infrastructure security as cyber threats and network breaches get more complex. A popular unsupervised machine learning method called K-Means was utilized to classify related network behavior patterns and differentiate legitimate traffic from possible dangers. The KDD Cup dataset, a common benchmark for intrusion detection tasks, was the main focus of the solution. Network records were classified into clusters that represented different traffic behaviors using K-Means clustering. While abnormal or malevolent behavior appeared as outliers or as members of separate clusters, normal behavior usually formed compact, consistent clusters.

The algorithm's performance was assessed using a number of measures, such as conditional entropy, purity, precision, recall, and F1-score. The findings showed that K-Means can detect anomalous traffic patterns with a respectable level of precision, particularly when it comes to differentiating between high-volume attacks like DoS and Probe. However, because of their rarity and resemblance to regular traffic, it demonstrated shortcomings in identifying low-frequency or covert attacks like R2L (Remote to Local) and U2R (User to Root).

K-Means' computational efficiency and scalability are two of its main advantages, which make it appropriate for near real-time analysis of massive network data. But the algorithm also has a number of drawbacks. Performance can be greatly impacted by the non-trivial decision of how many clusters to use (kkk). Furthermore, the intrinsic complexity of network traffic in the actual world might not be well suited to the assumption of spherical cluster geometries and equal cluster sizes.

6.1 Future Work

1) **Adaptive and Dynamic Clustering:** In dynamic network systems, K-Means may not be the best option because it presupposes a fixed number of clusters. Adaptive clustering techniques, such X-Means or G-Means, which automatically calculate the ideal number of clusters based on the data distribution, can be investigated in future research.
2) **Ensemble and Hybrid Models:** K-Means can produce higher detection rates when combined with other machine learning models. Unsupervised pre-clustering, for example, could be used in a hybrid model that combines K-Means with supervised classifiers (such as SVM or Random Forest) in order to lower noise and enhance classifier performance.

3) **Reducing Dimensionality and Feature Engineering:** Cluster separability can be improved by using dimensionality reduction methods such as t-SNE or PCA (Principal Component Analysis) and improving feature selection. K-Means and other distance-based algorithms are frequently less successful when high-dimensional features are present.
4) **Implementations of Real-Time and Streaming:** The method used now is batch-based. Real-time detection would be made possible by implementing online or streaming K-Means variants, like Mini-Batch K-Means, which would make their implementation in live network monitoring systems more feasible.
5) **Implementation and Real-World Assessment:** Deployment in real-world settings, like cloud or enterprise networks, is crucial for additional validation. This would enable the practical assessment of scalability, performance, and operational issues like model retraining and false positives.
6) **Managing Changing Dangers:** Future studies ought to concentrate on concept drift, the process by which what constitutes "normal" conduct evolves over time. Adapting to changing network patterns and new threats will need incorporating incremental learning into K-Means clustering.

7. References:

[1]*Song Wang - Machine Learning in Network Anomaly Detection, November,2021, https://creativecommons.org/licenses/by/4.0/

[2] J. Luo and S. Bridges, "Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection," International Journal of Intelligent Systems, May 2000. https://onlinelibrary.wiley.com/doi/10.1002/1098-111X(200008)15:8%3C687::AID-INT1%3E3.0.CO;2-X

[3] D. Barbara,´ J. C. S. Jajodia, L. Popyack, and N. Wu, "ADAM: Detecting intrusions by data mining," in Proceedings of the IEEE Workshop on Information Assurance and Security, Jun. 2001, https://dl.packetstormsecurity.net/papers/IDS/nids/ADAM-Detecting-Intrusions-by-Data-Mining.pdf

[4] P. Dokas, L. Ertoz, V. Kumar, A. Lazarevic, J. Srivastava, and P. N. Tan, "Data mining for network intrusion detection," in Proceedings of the NSF Workshop on Next Generation Data Mining, Nov. 2002. https://www.scirp.org/(S(i43dyn45te-exjx455qlt3d2q))/reference/referencespapers?referenceid=613261

[5] L. Ertoz, E. Eilertson, A. Lazarevic, P.-N. Tan, J. Srivastava, V. Kumar, and P. Dokas, Next Generation Data Mining. MIT Press, 2004, The MINDS - Minnesota Intrusion Detection System https://www-users.cse.umn.edu/~kumar001/papers/nsf_ngdm_2002.pdf

[6] M. Esposito, C. Mazzariello, F. Oliviero, S. P. Romano, and C. Sansone, "Evaluating pattern recognition techniques in intrusion detection systems." in Proceedings of the 5th International Workshop on Pattern Recognition in Information Systems (PRIS) 2005, May 2005, https://www.scitepress.org/papers/2005/25752/25752.pdf

[7] Hong Zhang - Atlantis Press, Improved K-means Algorithm Based on the Clustering Reliability Analysis, ISCI 2021 https://www.atlantis-press.com/article/17709

[8] Azhar Rauf, "Enhanced K-Mean Clustering Algorithm to reduce time complexcity, April,2023 https://www.researchgate.net/publication/273140830

[9] Liangchen Chen - An improved density peaks clustering algorithm based on grid screening and mutual neighborhood degree for network anomaly detection, April,2022, https://doi.org/10.1038/s41598-021-02038-z

[10] Mohammad Ali Moni - An Improved K-means Clustering Algorithm, June,2022, https://doi.org/10.1007/s40745-022-00428-2

[11] Thwaini MH. Anomaly Detection in Network Traffic using Machine Learning for Early Threat Detection. Data and Metadata. May 2022 . https://doi.org/10.56294/dm202272

[12] Lavanya Gundeoboina - Network Anomaly Detection Using Machine Learning, Apr,2022, https://www.ijarst.in/public/uploads/paper/875021682041527.pdf

[13] Purnendra Kumar - Network Traffic Analysis and Prediction using Machine Learning, Aug,2023, https://ijrpr.com/uploads/V4ISSUE8/IJRPR16324.pdf

[14] Oviya G, Network Traffic Anomaly Detection Using ML, April,2024, https://www.doi.org/10.58257/IJPREMS33135

[15] Manoharan Premkumar, K-Means Clustering algorithm for data clustering, June,2024 https://doi.org/10.1038/s41598-024-55619-z

[16] Kavitha D, Ramalakshmi R .,"Machine learning-based DDOS attack detection and mitigation in SDNs for IoT environments" .,Journal of the Franklin Institute 361 (2024) 107197.