An ensemble of Clustering Algorithms Using Different Distance Metrics for Network Traffic Analysis of Companies

Bissarinov Baituma[1], Valerii Lakhno [2], Valeriy Kozlovskyi[3], Denys Redko[*4], Alona Desiatko[4], Valentyn Yaremyvh[4]
[1]Department of Information systems. Al-Farabi Kazakh National University, Almaty, Kazakhstan
[2]Department of Computer systems and networks, National University of Life and Environmental Sciences of Ukraine, Kyiv, Ukraine
[3]Department of Information Protection System, National Aviation University, Kyiv, Ukraine
[4]Department of Software Engineering and Cybersecurity, State University of Trade and Economics, Kyiv, Ukraine
E-mail: bbaituma@gmail.com, lva964@nubip.edu.ua, valerii.kozlovskyi@npp.nau.edu.ua, d.redko@knute.edu.ua, desyatko@gmail.com, v.yaremych@knute.edu.ua
[*]Corresponding author

Keywords: big data, network traffic analysis, k-means, collective clustering, bayesian approach, co-associative matrix, distance metrics

Received: August 6, 2025

The network traffic analysis problem in large companies utilizing Big Data is considered. An ensemble of clustering algorithms based on Bayesian probability updating employing various distance metrics and adaptive weighting is proposed. An exponential dependence was applied in the weight calculation to enhance differentiation between algorithms. This enhanced the method's sensitivity to the quality of individual models. The developed approach was tested on open datasets CIC-IDS2017, UNSW-NB15, and CTU-13. The results demonstrated a consistent improvement in clustering quality, with ARI and NMI values reaching 0.78 and 0.75, respectively. The result surpasses the performance of baseline methods (K-means, DBSCAN, classical ensembles). The proposed method demonstrated linear scalability and is applicable for analyzing high-volume corporate network traffic. The results obtained confirmed the practical value of integration into monitoring and anomaly detection systems.

Povzetek: Opisan je ansambelski pristop h gručenju omrežnega prometa, ki združuje več algoritmov z različnimi metrikami razdalje ter Bayesovo prilagajanje uteži. Metoda izboljša kakovost gručenja in stabilnost na velikih podatkovnih zbirkah ter je primerna za analizo korporativnega prometa.

1 Introduction

Big Data finds application in various IT processes, such as those related to network traffic analysis for large companies, optimization and scaling of corporate network structure, extraction of information from web resources and other tasks to identify patterns in areas with extensive use of variant data that need to be structured, classified and analyzed in order to improve or optimize the business processes of companies [1], [2], [3], [4]. This study examines the use of a clustering algorithm ensemble, comprised of K-means algorithm variations distinguished by their distance metric between objects. In this case, when talking about the distance metric between objects for company network traffic analysis and optimization, it is assumed that the objects of the original data set will include a large set of elements, they may be data packets, where each packet passing through the network is considered as an object.

Then the distance metric will determine the degree of similarity or difference between different packets based on their characteristics such as IP addresses, ports, protocol, packet size and timestamps [5], [6]. The objects can also be network connections consisting of multiple data packets between certain IP addresses. In this case, the distance metric will take into account characteristics such as connection duration, amount of data transferred, packet rate and protocol types.

Alternatively, the objects may also be users or devices generating network traffic, in which case the distance metric will compare the behavior by considering the amount of data transferred, types of requests, time of activity, and/or geographic location of the subscriber. Traffic segments, such as specific time intervals or specific types of traffic (e.g., web traffic, email traffic, P2P traffic), can be additionally mentioned, which can also be considered as objects of analysis, and the distance metric in this case will determine whether different traffic segments are similar or different based on their characteristics.

As shown in [7], [8], [9], [10], [11], the use of different distance metrics will allow cluster analysis (hereinafter referred to as CA) algorithms to take into account the peculiarities of different types of objects and provide a detailed and relevant analysis of network traffic, which is important for the tasks of optimization and scaling of the corporate network structure. Approach, will enable a

better understanding of network behavior and more efficient allocation of resources required for effective operation of IT business processes of companies.

2 A review of prior research

A large number of CA methods and algorithms have been developed. The results of these studies were presented by the authors in [1] – [11], and in [12] – [20] a detailed review of the advantages and disadvantages of the considered approaches used in CA of company traffic data was conducted. Note that the results of grouping can vary significantly depending on the choice of the feature system, proximity measures, the choice of initial conditions, the order of objects, and the parameters of the algorithm, as shown in the analysis of these publications in classical algorithms for solving CA problems [19], [20]. Therefore, we believe that the efficiency of CA by applying the ensemble approach, which consists in building a set of clustering based on a variety of algorithms or one algorithm with different parameters and the final clustering based on them. For example, if network traffic data is available, then we can apply several KA algorithms (k-means, agglomerative clustering and DBSCAN), then calculate similarity metrics for each result and assign a weight-where ways to partition the traffic data set into clusters [21]. The final consensus is achieved through iterative algorithms that minimize the disagreement between the results [22]. Alternatively, different KA algorithms can be employed and for each pair of objects (e.g., IP addresses) determine how often they fall into the same cluster, and then the results are averaged and based on the resulting co-association matrix, KA algorithms (e.g., spectral clustering) are used to obtain a final consistent partition. As additional options, one can consider, mixture models of distributions. Data clustering can be leveraged by assuming that network traffic data follow certain distributions and using an EM algorithm to estimate the parameters of these distributions.

Several conceptual approaches to constructing ensemble solutions in CA are described in the scientific literature [17], [19], [21] – [26]. The first approach was to achieve consensus. That is, ensuring the highest possible degree of consistency with the results of individual CA algorithms is necessary. The second approach involved computing co-associative matrices (adjacency matrices). These matrices determine how frequently a pair of objects appears in the same cluster under different partitioning.

In recent years, research activity in the field of ensemble clustering and network traffic analysis has shifted toward adaptive and weighted methods [27], [28]. These methods, including those addressed in [27]–[30], consider the quality of individual ensemble members when constructing co-associative matrices and forming the final consensus. Recent works [31]–[32] have proposed adaptive weighting of co-associations and element-wise strategies for reinforcing the contribution of 'good' partitions. Bayesian formalizations of ensembles, as presented in [6], [14], [28], also provide for weight updates as new data arrives. Concurrently, [6], [25], [30], [34] are developing deep-clustering and hybrid (semi-/self-supervised + ensemble) methodologies. Despite the availability of a broad spectrum of cluster analysis methods, including ensemble approaches and advanced deep-clustering solutions [30], [32], [34], several unresolved challenges remain. Thus, most existing ensembles either rely on averaging the co-associative matrix. Additionally, the quality of individual algorithms is not taken into account. Alternatively, fixed weights are used, which do not adapt to the changing characteristics of traffic. Furthermore, although works on deep and hybrid clustering analysis [30], [32], [34] demonstrate good results, they are designed for specific tasks. This includes image processing or the analysis of encrypted traffic. However, interpretability and universality remain limited when applied to heterogeneous corporate network data. The reviewed literature lacks solutions that, simultaneously, provide adaptive updating of algorithm weights, amplification of their quality differences, and reproducibility of results on real-world network benchmarks.

The objective of this study was to develop an ensemble method for clustering network traffic that eliminates the aforementioned shortcomings. The study proposes the integration of a Bayesian approach for adaptive updating of algorithm weights and the use of an exponential mechanism to enhance distinctions among them. The method was evaluated on public benchmarks (CIC-IDS2017, UNSW-NB15, CTU-13) compared with baseline algorithms (K-means, DBSCAN) and traditional ensembles. Thus, this article seeks to address a gap in the literature and to justify the practical applicability of advanced ensemble approaches for Big Data analysis in corporate networks.

3 Methods and models

3.1 An ensemble of weighted algorithms for cluster analysis of heterogeneous data

Regarding analyzing network traffic for a large company, we can talk about heterogeneous data. Heterogeneous data refers to differences in the types and formats of data that are collected and analyzed in a corporate network. In addition to those mentioned above, this data may include: server logs; traffic data; sensor data (e.g., information from various network sensors such as IDS/IPS systems that detect suspicious activity on the network); user data; routing data; and others. These diverse types of data require different approaches to analysis and are often heterogeneous in nature, which makes them heterogeneous. To effectively analyze and process such data, it is advisable to use an ensemble of algorithms that can work with different metrics and take into account the specifics of each type of data to obtain a completer and more accurate picture of network traffic and its optimization.

Let us introduce the following notations to formalize the proposed method of building a collective solution that takes into account the weights of different algorithms for clustering the company's network traffic data.

Let us consider a set of objects $S = \{o^{(1)}, \ldots, o^{(N)}\}$ IP addresses, ports, protocol, packet size, server logs, traffic data, data from network sensors (e.g., IDS/IPS systems), user data, routing data, and so forth, randomly and independently selected from the overall population. It is necessary to partition these objects into a specified number of *C* clusters. A clustering quality criterion will be employed for the partitioning. Each object will be described by a set of real-valued variables. $X_1, \ldots, X_n$. This will be illustrated with several examples. Although IP addresses are typically represented in text format, they can be converted into numerical values for analysis. For instance, an IPv4 address can be represented as a 32-bit integer, which enables its use in computations. Port numbers are integers within the range from 0 to 65535, which makes them inherently suitable as real-valued variables for analysis. Network layer protocols (such as TCP and UDP) can be encoded using numerical values. TCP can be represented as 1 and UDP as 2, which enables the use of this data in numerical algorithms. Packet size is a real-valued variable indicating the number of bytes in a packet; this information is used for network performance analysis and anomaly detection. Logs contain a variety of information, including timestamps, status codes, response sizes, and so on. These data can be transformed into a set of numerical features (for example, timestamps into seconds or minutes, status codes into numerical values, etc.). Such data may include various metrics, such as volume of transmitted data, number of packets, and transmission time, which can be represented as real-valued variables. Information from network sensors (e.g., IDS/IPS systems) is generated as security event data, which may include timestamps, attack types, severity levels, and other parameters that may be numerical. User data may include user identifiers, session count, session duration, and other metrics that can be transformed into numerical values. Routing metrics, such as route cost, transit time, and similar parameters, can be represented as numerical values. These and other objects are converted into a set of real-valued variables to enable the application of clustering algorithms and other data analysis methods. Real-valued variables enable the straightforward application of mathematical operations and algorithms, making it possible to uncover hidden patterns and aiding in the development of more effective methods for analysis and processing of data based on cluster ensemble solutions.

The vector of variables is denoted for an (o) object by $x = x(o) = (x_1, \ldots, x_n)$. Here $x_j = X_j(o),\ j = 1, \ldots n.$ That is, the vector of variables for each (o) object will be a numerical description of (o), where each variable corresponds to a particular characteristic or feature of the object. For example, the IP address 192.168.1.1 can be represented as four numerical values. Accordingly, 192, 168, 1, 1. A port number, e.g., 443 for HTTPS is represented by a single numeric value. The packet size, 1500 bytes is represented by a single numeric value. And so on. Then, $x_N -$ is a $\left(x\left(\mathrm{o}^{(1)}, \ldots, \mathrm{o}^{(N)}\right)\right)^T$ data table. This table contains information about all the (o) objects. For example, IP addresses, ports, protocols, packet sizes, and so on, where each row represents a vector of variables for one (o).

In the context of a complex network topology within a large enterprise, there may exist a hidden variable Y. When we speak of a hidden (directly unobservable) variable $Y \in (1, \ldots, CL)$, which determines the assignment of each object to a specific class (CL), we imply that each object in our dataset belongs to one of several possible classes. These classes are not explicitly evident in the data and must be inferred based on the observable characteristics of the objects. As demonstrated in [21], [26], a class (or cluster) is characterized by a conditional distribution. This is reasonable, as it makes it possible to describe how the values of the features of objects belonging to a given class are distributed. That is, each class or cluster has unique characteristics. These differences are captured by conditional distributions. Therefore, it is correct $p(x|Y = cl) = p_{cl}(x),\ cl = 1, \ldots, CL.$ Conditional distribution $p(x|Y = 0)$ may indicate that ports 80 and 443 are more frequently used for HTTP/HTTPS, and that packet sizes follow a normal distribution with specific parameters. The conditional distribution $p(x|Y = 1)$ may be characterized by the frequent use of unusual ports, higher variance in packet sizes, and specific IP addresses that are often involved in attacks. Understanding conditional distributions will facilitate the relevant assignment of new objects to specific classes. Furthermore, for the given examples, this will enable the identification of deviations from normal behavior. This is essential for network security tasks, as well as for optimizing the network by improving routing processes and resource utilization.

Let us assume that each object is assigned to a class based on a priori probabilities. $P_{cl} = P(Y = cl),\ cl = 1, \ldots, CL.$ Here $\sum_{cl=1}^{CL} P_{cl} = 1.$ This means that prior to data analysis, there are already assumptions regarding the probabilities of objects belonging to various classes. That is, P_i denotes the a priori probability of an object belonging i to a specific cluster (class) prior to considering new data. These a priori probabilities are based on prior knowledge or data about the distribution of objects among the classes. Thus, for our task, this may imply that there are already initial assumptions regarding the types of network activities (or classes) to which each object may belong. (o). These probabilities may be based on historical data related to the company's network traffic, statistics on the occurrence of specific events, or other relevant information. For instance, we may assume that 70% of the traffic is associated with normal employee activity, 20% with automated systems and servers, and 10% with suspicious or anomalous activities. These assumptions are helpful in data analysis and clustering, as they specify the initial probabilities of objects belonging to various classes. This, in turn, enables the use of probabilistic methods for more accurate data analysis and processing. A priori probabilities serve as the starting point for determining to which class each object may belong and are used within the probabilistic data generation model for further analysis. In accordance with

$p_{cl}(x)$ we will define the value (x) independently for each (o). Next, for a pair of objects selected at random, for example, objects $a, b \in s$, we define their correspondence to the indicator function $I(\cdot)$ [21], [26]. That is, the value $H = I\big(Y(a) \neq Y(b)\big)$. Here $I(true) = 1,\ I(false) = 0$. Or, in other words, $\delta_{ij} = 1$, if the objects i, j fall into the same cluster according to the selected algorithm; $\delta_{ij} = 0$ otherwise.

Let us introduce the $P_H = P[H = 1|x(a),x(b)]$ notation, which describes the probability of the a, b event. Moreover, these events belong to different classes, with known$x(a)$ and $x(b)$. Then, based on the above, we can write the following expression for calculating P_H:

$$P_H = 1 - \sum_{cl=1}^{CL} \frac{p_{cl}(x(a))p_{cl}(x(b))P_{cl}^2}{p(x(a))p(x(b))}, \quad (1)$$

where $p\big(x(o)\big) = \sum_{cl=1}^{CL} p_{cl}\left(x(o)\right)P_{cl}, \quad o = a, b.$

Indicator function $I(\cdot)$ (indicator) is a function that is used to determine correspondence to a specific condition [26]. For example, if two data packets have the same source IP address, the indicator function may take the value 1. If two network connections use the same port, the indicator function may take the value 1. If two data packets use the same network protocol (for instance, HTTP or FTP), the indicator function may take the value 1. The indicator function $I(\cdot)$ facilitates the classification of objects and the analysis of their membership in different classes. (CL). In the context of clustering, this enables the consideration of how similar objects are to each other based on specific features, which is appropriate for assessing clustering quality and constructing cluster models for heterogeneous data.

When we use an ensemble of CA algorithms — $\mu_1, \mu_2, \ldots, \mu_M$ for example, K-means, Hierarchical Clustering, DBSCAN, Mean Shift, Gaussian Mixture Models (GMM), Spectral Clustering, Agglomerative Clustering, and OPTICS (Ordering Points To Identify the Clustering Structure)—we may obtain different variants of object set partitioning s into clusters. The number of clusters for each variant may differ, as different algorithms can group data in various ways. For example, suppose we have a dataset of company network traffic, including IP addresses, ports, protocols, and packet sizes. To analyze and optimize company traffic, various algorithms from the toolkit can be used for corresponding sub-tasks. For example, K-means can be used to group traffic by similar parameters, such as ports and IP addresses. DBSCAN can be applied to identify dense regions of traffic and detect anomalies. Hierarchical Clustering enables the creation of a hierarchical cluster structure, which can assist in identifying subclusters. By utilizing multiple algorithms and comparing their results, a more comprehensive understanding of the network traffic structure can be achieved, revealing hidden patterns and ultimately facilitating network optimization and enhancing security. Since the numbering of classes is irrelevant, it is more convenient to consider the equivalence relation. The equivalence relation enables us to determine whether two arbitrarily selected pairs of objects belong to the same class or to different classes. We define, in accordance with [21], for the pair a, b the value $r_m = I[\mu_m(a) \neq \mu_m(b)]$. As an example, suppose we need to consider two objects a and $b$representing network packets with specific characteristics, for example: 1) object a: IP address 192.168.1.1; port 80; protocol HTTP; packet size: 1500 bytes; 2) object b: IP address 192.168.1.2; port 443; protocol: HTTPS; packet size: 1500 bytes. Suppose we have two cluster analysis algorithms: μ_1 and $\mu_2$1) algorithm μ_1 divides the data by IP address and port; 2) algorithm μ_2 divides the data into clusters by protocol and packet size. For each algorithm, we determine whether the objects a and b fall into the same cluster. For the algorithm μ_1 objects a and b will be assigned to different clusters, since their IP addresses and ports differ. Then $r_1(a, b) = 1$. For the algorithm, μ_2 the objects will be assigned to different clusters, since their protocols differ. Then $r_2(a, b) = 1$. By employing a set of CA algorithms and defining indicator functions $q_m(a, b) = 1$ for each pair of objects, we can construct a set of partitions of the objects into clusters, which enables more flexible and accurate analysis of the data, taking into account various criteria and metrics.

Since the task of finding the optimal partitioning of network traffic according to a specified criterion is of exponential complexity, approximate iterative algorithms are also employed in practice. At each step, these algorithms modify the current partitioning, seeking a local improvement in quality. The algorithm operations was regulated by user-defined parameters.

In [21], [26], the concept of a 'constant conditional probability of a correct solution' is discussed— q_m. In our case, this means that for each algorithm $\mu_m$used in the ensemble (or collective decision), the probability of correctly merging or splitting a pair of objects remains constant and does not depend on the specific pair of objects. If the algorithm employs the Euclidean metric to measure the distance between objects, its accuracy will be consistent across all pairs of objects. In other words, the parameter q_m is the probability that the algorithm correctly groups two objects into a single cluster if they indeed belong to the same cluster, or correctly separates them into different clusters if they belong to different clusters. The value q_m makes it possible to assess how effectively the algorithm performed the clustering task. The higher the value, q_m, the more reliable we consider the algorithm. The condition $q_m > 0{,}5$ will be referred to as the 'weak learning' condition, meaning that the algorithm makes decisions better than a random choice. This is necessary for constructing an ensemble (collective) of algorithms, as it guarantees that each algorithm in the ensemble contributes positively to the overall solution. Let us illustrate this with an example. Suppose there are server logs and data from network sensors. Let us utilize the DBSCAN algorithm. DBSCAN determines that a particular log and sensor data belong to different clusters. If $q_m = 0{,}7$, this means that the probability that DBSCAN correctly determines that these data belong to different clusters is 70%.

Parameter q_m is fundamental for understanding and evaluating the effectiveness of each algorithm in the ensemble; see Fig. 1. It demonstrates the accuracy of each algorithm's decisions and helps determine how each will impact the overall clustering result. Figure 1 shows a schematic diagram of how algorithms are combined into an ensemble, where each algorithm is assigned, a weight reflecting its significance and accuracy for a specific data type. The final clustering decision is determined by the weighted voting of all algorithms, which enables the contribution of each to be considered and enhances both the accuracy and reliability of the results.

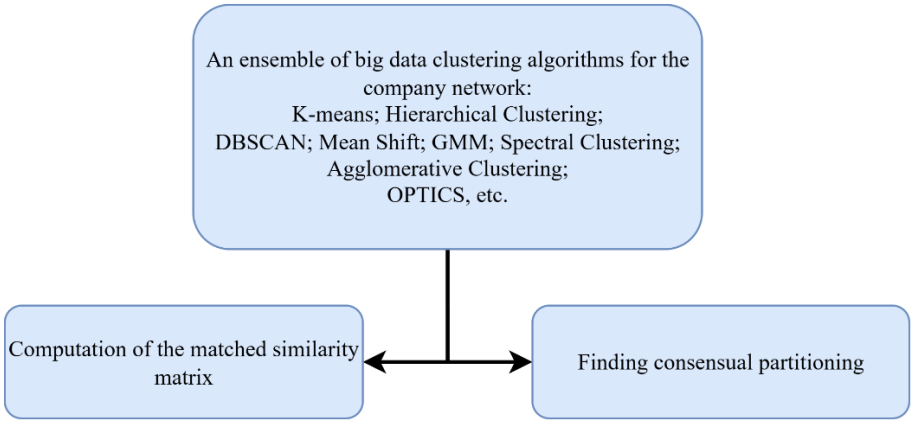


Figure 1: Interaction of algorithms in collective solving.

One of the main challenges of CA Big Data for network traffic in large companies is the ambiguous interpretation of results. Clustering algorithms based on various approaches may yield different results, which complicates decision-making. To enhance the methods proposed in [1], [2], [17], [21], [26], which consider the behavior of each algorithm under various conditions, a probabilistic model of ensemble pairwise classification with latent classes can be utilized. This model enables the weighting of each algorithm based on its performance under different conditions. Latent classes refer to hidden or implicit categories that are not directly observed but influence the behavior and characteristics of the data. Latent classes are hypothetical categories that help explain the structure of the data. For example, in network traffic analysis, these may include groups of users with certain behavioral patterns, types of devices, types of network attacks, and other hidden factors that influence network activity. For instance, when analyzing network traffic data, including IP addresses, ports, protocols, packet sizes, server logs, and data from network sensors, latent classes may represent various types of network devices (such as servers, workstations, and mobile devices) or types of network attacks (DDoS attacks, phishing, or intrusions).

In the probabilistic model of ensemble pairwise classification with latent classes, it is assumed that each object (for example, a network traffic record) belongs to one of the hidden classes. Such membership influences its probability distribution. For instance, a network traffic record belonging to the 'mobile devices' class may have distinctive characteristics that differ from records in the 'servers' class.

Note that determining the weights of clustering algorithms within the ensemble is a critical step for improving the accuracy and reliability of the CA. For this purpose, as demonstrated in [21], a probabilistic model of ensemble pairwise classification with latent classes can be employed. Let us consider the mathematical formulation of this model and the definition of the weights. For an ensemble of clustering algorithms composed of K-means variations, we propose the following methodology for weight calculation; see Figure 1.

At the first stage, we conduct an assessment of clustering quality using quality metrics [26], [27], [28]—ARI, NMI, and others.

For the second stage, we will formulate a hypothesis.

Hypothesis: Employing a probabilistic ensemble cluster analysis model with weights reflecting the reliability of classification results for each pair of objects will enhance the accuracy and robustness of traffic clustering in a large company. At this stage, a probabilistic model will be constructed for each clustering algorithm—comprising variations of K-means—taking into account its effectiveness in clustering data based on quality metrics and utilizing a Bayesian approach to update prior probabilities as new data become available.

In [11] and [21], the authors examined an ensemble of clustering algorithms, where each algorithm depends on a random vector. Ω. Clustering results may vary between different runs. Ω. Statistical dependence will indicate that the algorithm's decisions μ_m regarding the assignment of objects to clusters are associated with the true classes of these objects. In other words, if two objects in fact belong to the same class, then an algorithm constructed on rational grounds will, with high probability, assign them to the same cluster. Conversely, if the objects in fact belong to different classes, the algorithm will, with high probability, assign them to different clusters. An example follows. Let us suppose that we are analyzing network traffic. Suppose we need to group packets by connection type (HTTP, FTP, SSH, etc.) using variations of the basic K-means algorithm. The relevant variation of the K-means

algorithm may utilize various packet characteristics, such as source address, destination address, protocol type, packet size, and so on. In this scenario, statistical dependence would indicate that if two packets exhibit similar characteristics, the algorithm is highly likely to assign them to the same cluster, even if their random vectors differ. Ω. Conversely, if two packets have different characteristics, the algorithm is highly likely to assign them to different clusters, even if they have identical random vectors. Ω. As demonstrated in [21], [26], achieving statistical dependence is a relevant task in the development of a clustering algorithm ensemble, as it enables the algorithm to more accurately reflect the actual class structures within the data. However, statistical dependence does not guarantee perfect clustering. Indeed, even if the algorithm μ_m always correctly classifies pairs of objects with 'identical' characteristics, it may err when the characteristics of objects overlap between classes.

It can be argued that for each algorithm $\mu_m$included in the ensemble (collective decision), we can use the following formula to update the posterior probability. $P(A_i|D)$:

$$P(A_i|D) = \frac{P(D|A_i)P(A_i)}{\sum_{j=1}^{n} P(D|A_j)P(A_j)} \quad (2)$$

where $P(A_i)$ −the prior probability of the algorithm A_i ; $P(D|A_i)$ −probability of data observation D assuming that the algorithm is employed A_i. $P(D|A_i)$ calculated based on quality metrics [21]; $\sum_{j=1}^{n} P(D|A_j)P(A_j)$ −a normalization factor that ensures the posterior probabilities sum to one.

Let us illustrate this with an example. Suppose we have three variants of the K-means algorithm: 1) $\mu_1 = A_1$ −a K-means variant using Euclidean distance. This approach is suitable if, as part of the overall research objective, it is necessary to detect anomalous traffic. In many cases, anomalous traffic may indicate possible network attacks or malfunctions; 2) $\mu_2 = A_2 -$ A variation of K-means with cosine distance. For example, this variant of K-means is suitable for clustering employees based on the types of their network activity. In such a subtask, the goal is to group employees whose network usage profiles (e.g., visited websites, used applications) exhibit similar patterns. Cosine distance is beneficial as it enables the evaluation of angular similarity between activity vectors, regardless of their absolute magnitude, which is appropriate since employees may differ in the volume of network activity but demonstrate similar usage patterns. $\mu_3 = A_3 -$ A variation of K-means using the Minkowski distance, which is suitable for clustering network traffic sessions to detect anomalous behavioral patterns. The objective is to group network sessions by characteristics such as session duration, volume of transmitted data, and the number of requests to different nodes. Minkowski distance enables flexible adjustment of the influence of various dimensions, which is beneficial for capturing different types of anomalies. For example, this includes short but intense bursts of traffic or long-lasting but low-intensity connections.

Initially, the a priori probabilities are equal for all algorithms; thus, we can write: $P(A_1) = P(A_2) = P(A_3) = 1/3$. Accordingly, for a greater number of variational algorithms, the proportion will differ.

Using the approaches outlined in [19], [24] and having performed clustering, we can subsequently calculate the ARI and NMI metrics for each configuration. Let us assume that the following values have been obtained:

$$ARI_{A_1} = 0{,}8, \quad NMI_{A_1} = 0{,}75, \quad ARI_{A_2} = 0{,}6,$$
$$NMI_{A_2} = 0{,}65, \quad ARI_{A_3} = 0{,}7, \quad NMI_{A_3} = 0{,}7.$$

For assessment $P(D|A_i)$ We will employ the normalized sum of metrics.

$$P(D|A_i) = \frac{ARI_{A_i} + NMI_{A_i}}{\sum_{k=1}^{4}\left(ARI_{A_i} + NMI_{A_i}\right)}. \quad (3)$$

Thus, we obtain the following results for $P(D|A_i)$:

$$P(D|A_1) = 0{,}369, \; P(D|A_2) = 0{,}298,$$
$$P(D|A_3) = 0{,}333.$$

Next, we apply Bayes' formula to update the probabilities:

$$P(A_1|D) = \frac{P(D|A_1)P(A_1)}{P(D|A_1)P(A_1)+P(D|A_2)P(A_2)+P(D|A_3)P(A_3)} =$$
$$\frac{0{,}369\cdot\left(\frac{1}{3}\right)}{0{,}369\cdot\left(\frac{1}{3}\right)+0{,}298\cdot\left(\frac{1}{3}\right)+0{,}333\cdot\left(\frac{1}{3}\right)} = 0{,}369,$$
$$P(A_2|D) = 0{,}299,$$
$$P(A_2|D) = 0{,}333.$$

This approach, within the development of collective cluster-based solutions for Big Data analysis concerning network traffic issues in large corporations, may offer certain advantages, as the Bayesian approach enables adaptive updating of algorithm weights based on new data, which is essential under conditions of variable traffic. Furthermore, the use of quality metrics for updating probabilities will improve clustering accuracy, although this assertion requires experimental validation.

3.2 Conceptual diagram of the stages of ensemble clustering algorithms using different distance metrics for analyzing company network traffic

Unlike the studies [17], [18], [21], [26], which employ a simple averaging of the co-association matrix, we propose accounting for the weights of each object pair based on an exponential function. The sequence of steps for implementing this approach is outlined below.

The main idea is that, in our case, the weight of each algorithm in the ensemble depends on its performance, as determined by quality metrics such as ARI and NMI. We propose utilizing an exponential function, which allows accentuating the differences in the algorithm weights, even when the differences in quality metrics are minor. This may potentially make the ensemble CA (collective decision) method more sensitive to varying levels of performance. We employ the exponential dependence (4) in our analysis to amplify the differences in weights. The paradigm of this approach is based on the premise that minor changes in quality metric values lead to more substantial changes in weights. This is significant, as more

efficient algorithms should be assigned a considerably higher weight, given that their impact on the final result will be greater.

Then, taking the above into consideration, we can write:

$$w_i = \frac{exp(\alpha \cdot Q_i)}{\sum_{j=1}^{N} exp(\alpha \cdot Q_j)}, \quad (4)$$

where w_i −weight i −of the CA (in other words, w_i − weight i −of the CA in the ensemble, calculated based on its quality (ARI, NMI) and normalized so that the sum of the weights across all algorithms equals 1); Q_i −the corresponding quality metric (e.g., ARI or NMI) i −of the CA; α − a parameter that regulates the extent to which differences in weights are amplified (it can be selected experimentally; in other words, α - the exponential coefficient or 'hyperparameter regulating weight sensitivity); N −The total number of algorithms in the ensemble (collective decision).

Stage 1. Let us consider an example with three variations of the K-means algorithm, employing different distance metrics: Euclidean, cosine, and Minkowski. Suppose that the quality metrics for these algorithms (e.g., ARI) are equal to $Q_{\text{Eucl}} = 0.8$, $Q_{\text{Cos}} = 0.6$, $Q_{\text{Mink}} = 0.6$, respectively. These data have not been obtained experimentally, and are adopted conditionally for developing the proposed ensemble cluster analysis method [21], [26].

Step 1: Let us calculate the exponential weights for $Q_{\text{Eucl}} = 0.8$, $Q_{\text{Cos}} = 0.6$, $Q_{\text{Mink}} = 0.6$. And $\alpha = 5$.

Then

$$exp(5 \cdot 0{,}8) \approx 54{,}6;$$
$$exp(5 \cdot 0{,}6) \approx 20{,}09;$$
$$exp(5 \cdot 0{,}7) \approx 33{,}12;$$
$$\sum exp(5 \cdot Q) = 107.81.$$

Next, let us compute the normalized weights:

$$w_{\text{Eucl}} = \frac{54{,}6}{107{,}81} = 0{,}506;$$
$$w_{\text{Cos}} = \frac{20{,}09}{107{,}81} = 0{,}186;$$
$$w_{\text{Mink}} = \frac{33{,}12}{107{,}81} = 0{,}307.$$

Once the weights for each algorithm are known, we can then use them to form an average co-associative matrix, taking into account the weight of each pair of objects. For example, if a pair of objects is clustered with weight w_{Eucl}, then the contribution of this pair to the co-associative matrix will be proportional to this weight. We believe that the use of exponential dependence in the calculation of weights will allow us to more accurately account for differences in the performance of different algorithms, which is important in the tasks of analyzing network traffic of a large company. In parallel, it will improve the quality of collective clustering, making the results more reliable.

Stage 2: In the second step, we need to form a co-associative matrix using the weights calculated in the first step. The co-associative matrix reflects the probability that a pair of $o(i,j)$ objects is in the same cluster based on the results of all clustering algorithms in the ensemble. The inclusion of weights will allow us to take into account the degree of reliability of the results of each algorithm.

According to [26] we will use such a dependence to form the co-associative matrix:

$$C_{i,j} = \frac{1}{N}\sum_{k=1}^{N} w_k \cdot 1\big(A_k(i) = A_k(j)\big), \quad (5)$$

where is the value of the co-associative matrix for a pair of $o(i,j)$ objects; N − the number of algorithms involved in the collective solution (in the ensemble); w_k − the weight of the k −algorithm; $1\big(A_k(i) = A_k(j)\big)$ − the indicator function (in the general case$I(\cdot)$), which is equal to 1 if $o(i,j)$ are in the same cluster according to the results of the k −algorithm, and 0 otherwise.

Similar to the first step, consider a similar example with three variations of the K-means algorithm using different distance metrics: Euclidean, Cosine, and Minkowski.

$$w_{\text{Eucl}} = \frac{54{,}6}{107{,}81} = 0{,}506;$$
$$w_{\text{Cos}} = \frac{20{,}09}{107{,}81} = 0{,}186;$$
$$w_{\text{Mink}} = \frac{33{,}12}{107{,}81} = 0{,}307.$$

Suppose we have three objects (o_1, o_2, o_3) and the clustering results for each metric are as follows, see Table 1.

Table 1: Illustrative examples with clustering results for each metric

Variations of the K-means algorithm using different distance metrics	**Cluster 1**	**Cluster 2**
Euclidean	(1, 2)	(3)
Cosine	(1)	(2, 3)
Minkowski	(1, 3)	(2)

Then, for each pair of objects, $o(i,j)$ we compute $C_{i,j}$Pair (1,2) $C_{1,2} = \frac{1}{3}(0{,}506 \cdot 1 + 0{,}186 \cdot 0 + 0{,}307 \cdot 0) \approx 0{,}169$. Next, pair (1,3) $C_{1,3} \approx 0{,}102$. Pair (2,3) $C_{2,3} \approx 0{,}062$.

After forming the co-associative matrix, the final clustering method can be applied next. We can use hierarchical clustering, to obtain the final clusters, which will take into account the results of all the algorithms in the ensemble.

Hierarchical clustering is a powerful tool for creating final clusters after the formation of the co-associative matrix, because it allows you to create dendrograms that provide analysts with a visual representation of the nesting of hierarchies of clusters obtained in the process of analyzing data on company traffic, so you can see how the clusters are formed and how they are interconnected. For example, for our task for a large company network, we can see how individual "small" traffic groups are clustered into larger categories (e.g., individual applications can be grouped into more general usage categories such as "social networking" or "business applications"). Also note that hierarchical clustering does not require a predetermined number of clusters. This is important when working with Big Data, where the number of clusters may be unknown. When analyzing the network traffic of a notional company, you may find that the number of clusters required to adequately separate data may vary depending on the time of day or season. And this makes hard limits on the number of clusters ineffective.

A co-associative matrix created from the results of several clustering algorithms allows hierarchical clustering to take into account the entire population of data, providing more accurate clusters. Moreover, another combination of algorithms is also possible. In this case, if different algorithms in the ensemble gave different partitions, hierarchical clustering will be able to effectively integrate these partitions to get a more consensus and accurate representation of the structure of the data.

Hierarchical clustering methods, such as agglomerative clustering, can be optimized for Big Data applications, in particular when efficient methods of data storage and processing are employed. This is particularly effective in scenarios where distributed computing and optimized algorithms enable the application of hierarchical clustering to the traffic of large corporate networks, making it possible to process substantial data volumes within an acceptable timeframe.

Stage 3 – In the third step, using the generated co-associative matrix, we need to perform the final clustering. This can be done using a clustering algorithm that works with co-associative matrices.

Assume that we have three objects - three objects (o_1, o_2, o_3) and a co-associative matrix, calculated in the previous step:

$$C = \begin{pmatrix} 1 & 0{,}169 & 0{,}102 \\ 0{,}169 & 1 & 0{,}062 \\ 0{,}102 & 0{,}062 & 1 \end{pmatrix}.$$

Hierarchical clustering is performed as follows. We begin with each object in a separate cluster. At each step, we merge the two clusters with the highest value in the co-associative matrix. We repeat this process until all objects (o_1, o_2, o_3) are merged into a single cluster or the specified number of clusters is reached. For example, we merge objects 1 and 2, as $C_{1,2} = 0{,}169$ –this represents the maximum value, excluding the diagonal elements. Next, we merge the resulting cluster {1,2} with object 3, as $C_{1,3} = 0{,}102$ and $C_{2,3} = 0{,}062$. As a result, we obtain two clusters: cluster 1 – (o_1, o_2); cluster 2 – (o_3).

The use of the co-associative matrix and the hierarchical clustering algorithm allows us to consider the weighting coefficients determined by the reliability of each algorithm in the ensemble, enabling a more accurate partitioning of the data into clusters. The research conducted makes it possible to formulate the following stages of the operation of the ensemble of clustering algorithms employing various distance metrics for network traffic analysis in companies, see Fig. 2. Furthermore, based on the above considerations, the algorithm for constructing the cluster ensemble (collective decision), as presented in [18], [21], [26], has been clarified, see Fig. 3.

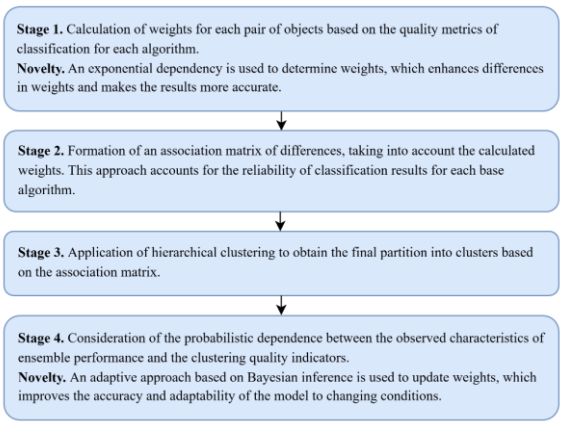


Figure 2: Conceptual diagram of the stages of the ensemble of clustering algorithms using different distance metrics to analyze company network traffic.

Step 1. Calculation of weights for each pair of objects based on the classification quality metrics of each algorithm. An exponential dependency is applied to determine the weights. This will enhance the differences in weights, resulting in increased accuracy.

Stage 2. Construction of a co-association difference matrix with consideration of the calculated weights, which enables the reliability of the classification results of each base algorithm to be taken into account.

Stage 3. Application of hierarchical clustering to obtain the final partitioning into clusters based on the co-association matrix.

Stage 4. Accounting for the probabilistic dependence between the observable characteristics of the CA ensemble's operation and the clustering quality metrics.

We provide brief clarifications for certain blocks of the algorithm depicted in Figure 3. We also present the description in the form of pseudocode (Algorithm 1).

Algorithm 1. Weighted Ensemble Clustering with Bayesian Updating

Input data:

$X = \{x_1, x_2, ..., x_n\}$– a set of objects (network sessions, packets, connections, etc.);

$A = \{A_1, A_2, ..., A_m\}$– a set of clustering algorithms (variations of K-means with different distance metrics, DBSCAN, and others);

α – exponential amplification coefficient;

$Q(.)$ – clustering quality evaluation function (ARI, NMI).

Output data:

Final partitioning of the set X into clusters.

Algorithm steps:

Data clustering. For each algorithm $A_i \in A$, perform clustering of the set X and record the cluster labels.

Quality evaluation. For each algorithm, calculate quality metrics. q_m.

Weight calculation. Calculate the weight w_i of each algorithm using the exponential formula.

Construction of the co-associative matrix. For each pair of objects, compute $C_{i,j}$.

Weight update (Bayesian rule). For each algorithm, recalculate the posterior probabilities based on the computed metrics and the co-associative matrix.

Final clustering. Apply a method (hierarchical clustering or another approach) to the co-associative matrix C to obtain the final partitioning into clusters.

Objects refer to individual elements of network traffic, such as data packets, sessions, IP addresses, requests, and so on. Features (or attributes) are the characteristics or properties of these objects.

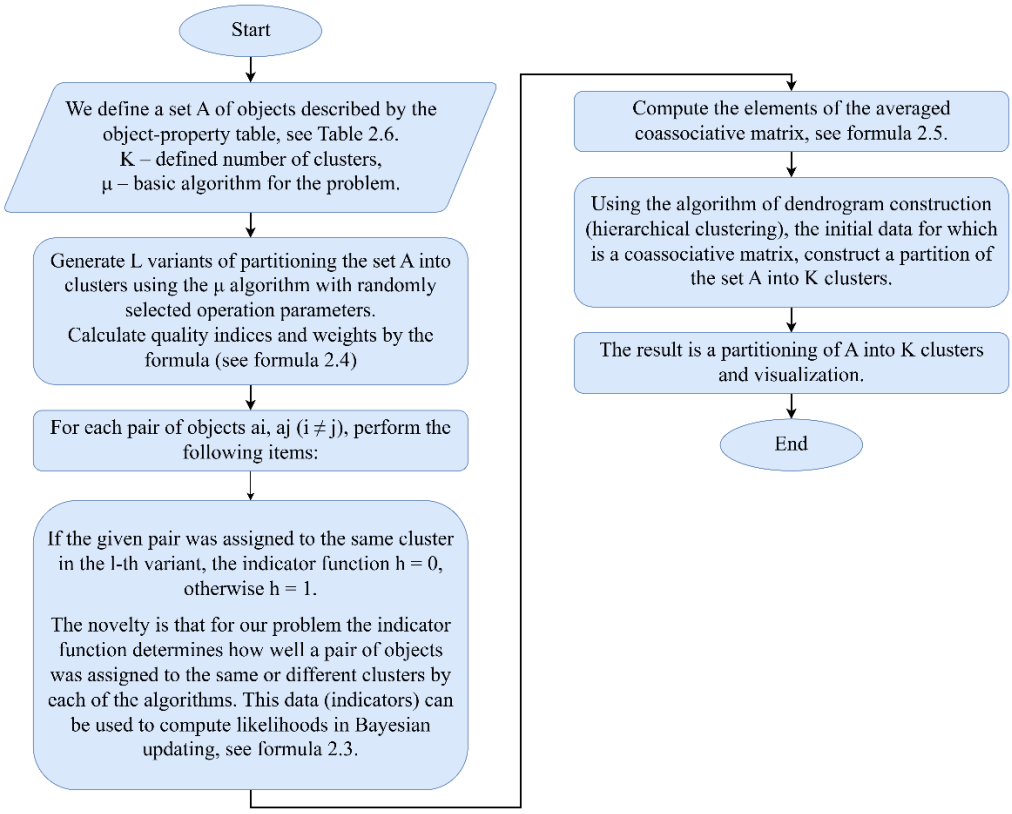


Figure 3: Refined enlarged algorithm for building a cluster ensemble (collective solution) for analyzing network traffic of companies.

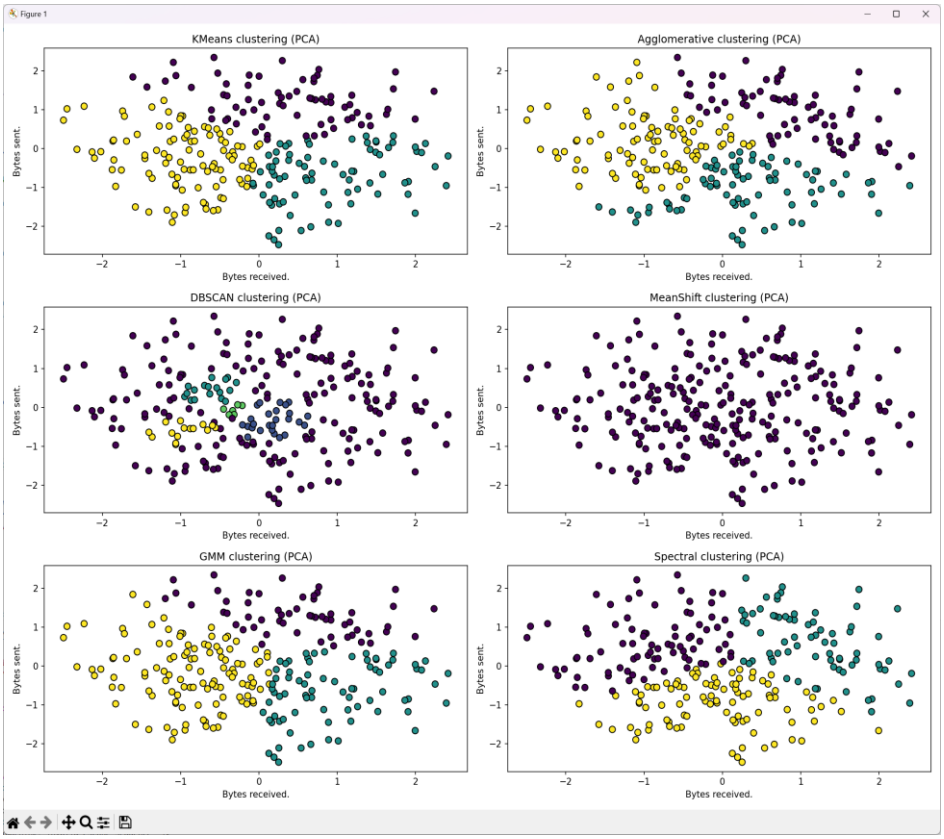
4 Software implementation

Using the proposed algorithm, we can consider the network traffic analysis problem for such a set of data as shown in Table 2.

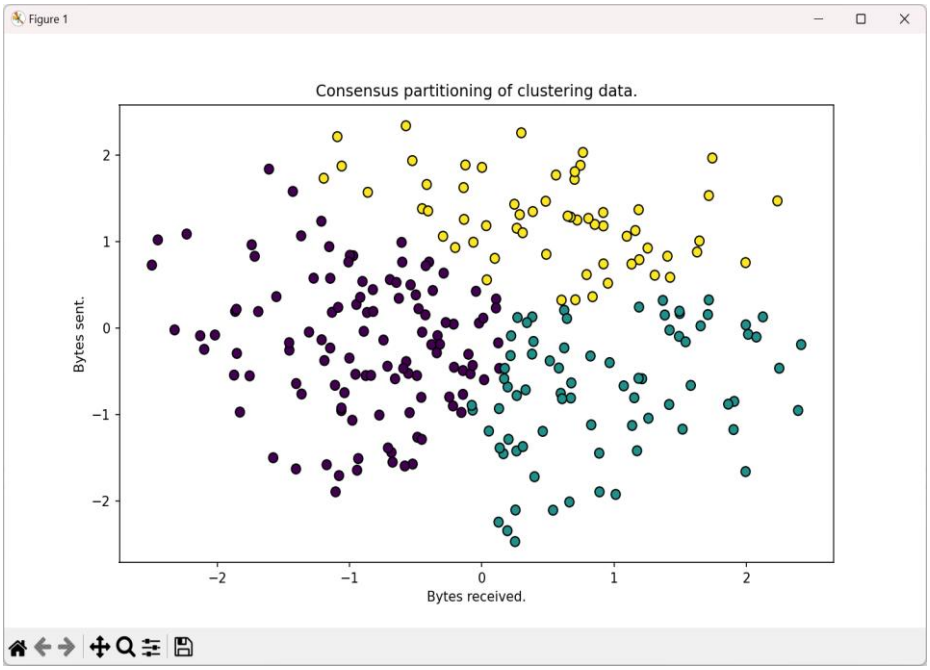
Suppose we have session data described by various features (IP addresses, ports, data volume, etc.). We apply an ensemble of clustering algorithms, implemented in the Python programming language (VS Code interpreter), each of which forms its own clusters (see Fig. 4a). By applying each algorithm to the data, we obtain cluster labels, which are identifiers indicating the cluster assignment of each object. When a clustering algorithm is applied to a dataset, the algorithm partitions the data into clusters, and each object is assigned a label indicating its cluster. Next, according to the algorithm, we use an indicator function to determine whether pairs of objects are in the same cluster. At this stage, we use the results of the indicator functions to calculate the likelihoods for each algorithm and apply a Bayesian approach to update the prior probabilities of the algorithms. Then, we use the indicators and the updated probabilities for formation of the co-associative matrix, see Fig. 4b). Finally, hierarchical clustering can be applied to the co-associative matrix to obtain the final clusters.

Table 2: Examples of "Object - Property" set for the task of analyzing network traffic of a large company.

Object (Session)	Source IP	IP Destination	Source Port	…	…	…	…	…	Traffic Type
Session 1	192.168.0.1	192.168.0.10	12345	…	…	…	…	…	HTTP
Session 2	192.168.0.2	192.168.0.30	12346	…	…	…	…	…	HTTP
…	…	…	…	…	…	…	…	…	…
Session T	192.168.0.30	192.168.0.12	12355	…	…	…	…	…	SSH



(a) Using different clustering algorithms to analyze company traffic.



(b) Clustering with new features after Principal Component Analysis (PCA).

Figure 4: Implementation of the algorithm for building a cluster ensemble (collective solution) for analyzing network traffic of companies.

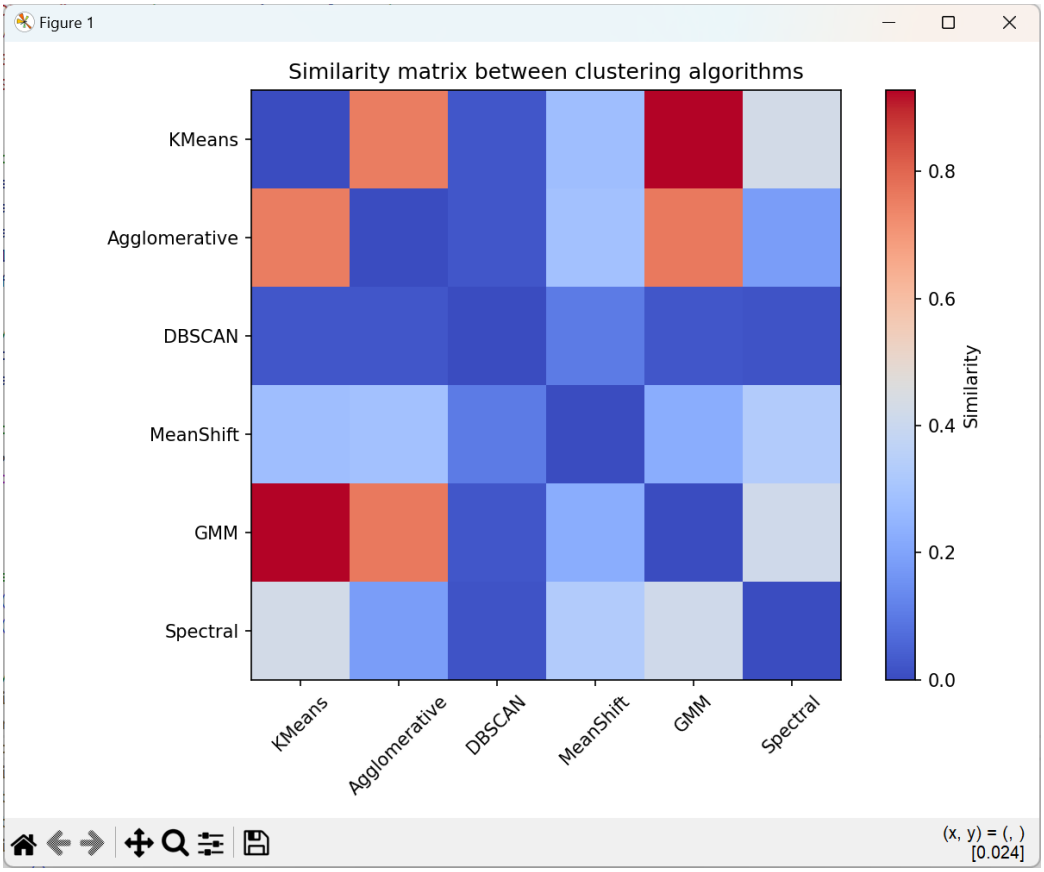


Figure 5: Visualization of the similarity matrix between clustering algorithms.

The approach proposed in the article enabled consideration of the quality of each algorithm (see Fig. 5) within the ensemble and allowed for adaptive updating of the weights based on new data, thereby improving overall clustering performance.

An additional series of experiments was conducted on publicly available network traffic benchmarks during the course of the research to demonstrate the reproducibility and evaluate the versatility of the proposed ensemble clustering method. These benchmarks are used in international studies on network security. The exclusive use of corporate enterprise data obtained during pilot deployment does not ensure comprehensive validation of the algorithm's resilience, since the internal traffic reflected the specific architecture, protocol profiles, and network activity schedule of the particular company. However, it did not encompass the full spectrum of contemporary attacks.

Verification was further supplemented with the following datasets, respectively [35], [36], [37]; see Table 3.

CIC-IDS2017 / CSE-CIC-IDS2018 (Canadian Institute for Cybersecurity) – representative traffic with simulated DoS, DDoS, botnet activity, SQL injection, and other modern threats [35].

UNSW-NB15 is traffic emulating real corporate networks with various applications and simultaneously incorporating nine types of attacks [36].

CTU-13 is a collection of real botnet traces, useful for evaluating the resilience of the ensemble to covert command channels [37].

These datasets enable the assessment of proposed algorithm's scalability and correctness in the event of a sharp increase in data volume, diversity of protocols, and the presence of 'normal/anomalous' traffic labels. For each dataset, the initial conditions were maintained: preliminary normalization, feature space construction, formation of local clusters using different metrics, and subsequent aggregation into a collective solution with Bayesian weight updating. This strategy ensures result comparability and enables evaluation of the ensemble's ability to adequately group previously unknown patterns of network activity.

Table 3: Comparison of clustering quality on open datasets.

Dataset	Method	ARI	NMI	Execution time*	Stability under α
CIC-IDS2017	K-means (Euclidean)	0,62	0,60	1,0×	–
	DBSCAN	0,65	0,63	2,3×	–
	Traditional ensemble	0,70	0,68	2,8×	Low
	Proposed method	0,78	0,75	3,0×	High
UNSW-NB15	K-means (Euclidean)	0,60	0,59	1,0×	–
	DBSCAN	0,63	0,61	2,2×	–
	Traditional ensemble	0,68	0,66	2,6×	Low
	Proposed method	0,76	0,74	2,9×	High
CTU-13	K-means (Euclidean)	0,57	0,55	1,0×	–
	DBSCAN	0,61	0,59	2,1×	–
	Traditional ensemble	0,66	0,64	2,4×	Medium
	K-means (Euclidean)	0,57	0,55	1,0×	–
Proposed method		0,73	0,71	2,7×	High
* Relative execution time: '1.0×' corresponds to the baseline K-means; All other values are presented in normalized units.					

As shown in Table 4, existing solutions demonstrate quality improvements on individual datasets. However, they exhibit certain limitations: fixed algorithm weights, high sensitivity to parameters, narrow specialization, or limited scalability in Big Data environments. These observations confirm a gap in the literature and substantiate the relevance of developing an adaptive ensemble with Bayesian weight updates and an exponential mechanism for emphasizing differences.

Table 4: Comparison of the proposed method with existing approaches for the network traffic clustering task.

Study	Method	Dataset	Metrics	Results	Limitations / Gaps
[30]	Deep Embedded Clustering (DEC)	CICIDS2017	ARI, Accuracy	High accuracy on small subsamples.	No adaptive weights; limited interpretability.
[32]	DBSCAN + Variational Autoencoder	UNSW-NB15	NMI, F1	Improvement compared to classical DBSCAN.	Parameter sensitivity; poor scalability.
[34]	Spectral Clustering Ensemble	CTU-13	Silhouette, NMI	Stable results on static data.	Does not account for traffic dynamics; fixed weights.

[35]	K-means Ensemble (different initializations)	KDD99	ARI, Purity	Enhancing cluster stability.	No Bayesian updating, weak performance on large-scale data.
[37]	Hybrid CNN + Clustering	CICIDS2017	Accuracy, F1	Produces strong results on encrypted traffic.	High computational costs, low versatility.
Proposed method	Ensemble with Bayesian weight updating and exponential boosting	CIC-IDS2017, UNSW-NB15, CTU-13	ARI, NMI, Runtime	ARI up to 0.78, NMI up to 0.75; Linear scalability.	No real-time support; requires adaptation for encrypted traffic.

The results obtained confirmed that the proposed ensemble with Bayesian weight updating and exponential boosting demonstrates an advantage over baseline methods in terms of both clustering quality (ARI, NMI) and stability under parameter variation. α. Stability is important when analyzing heterogeneous network traffic, as the sensitivity of algorithms to parameters may diminish the reliability of conclusions. The set of experiments conducted demonstrates that the methodology maintains its effectiveness across various datasets. It possesses significant potential for integration into practical network security monitoring systems.

5 Discussion of the findings

The research develops established ensemble (collective) cluster analysis approaches by introducing several improvements, which include: constructing a co-association difference matrix that accounts for the weights of base algorithms; utilizing an exponential function to intensify differences in weights; considering the probabilistic relationship between ensemble performance characteristics and clustering quality metrics; and applying a Bayesian approach to update prior probabilities based on new data. These innovations enable more accurate and reliable analysis and processing of Big Data network traffic in large enterprises.

Experiments on open benchmarks (see Table 3) confirmed that the proposed ensemble produces consistent results even beyond the original corporate datasets. The average ARI and NMI scores on CIC-IDS2017 and UNSW-NB15 exceeded 0.78 and 0.75, respectively, which is comparable to or exceeds the baseline methods (K-means, DBSCAN, traditional ensembles without Bayesian weighting). On the CTU-13 dataset, the method demonstrated increased cluster stability when varying the exponential amplification parameter α. A significant observation was the linear scaling of execution time with an increase in the number of packets. This result indicates the algorithm's suitability for analyzing high-volume network traffic within the infrastructure of a large enterprise.

Asymptotic analysis shows that the computational complexity of the proposed ensemble is comprised of the costs of the individual clustering algorithms and the construction of the co-association matrix. For K-means variations, it is equal to $O(n \cdot k \cdot t)$, where *n* is the number of objects, *k* is the number of clusters, and *t* is the number of iterations. The formation of the co-association matrix has a complexity of $O(n^2)$. This stage inherently supports parallelism and can be accelerated using distributed platforms (Spark, Hadoop). This ultimately confirms the practical viability of the proposed approach in Big Data environments.

In general, the results obtained in the course of the experiments demonstrate the possibility of integrating the developed solution into existing SIEM systems and substantiate its further application to early anomaly detection tasks, including those involving encrypted traffic.

6 Conclusions

This research has yielded the following results. Big Data is applicable in various processes related to the analysis of network traffic in large enterprises, tasks of optimizing and scaling corporate network structures, extraction of information from web resources, and other pattern identification tasks in domains characterized by intensive Big Data usage. These data require structuring and analysis to optimize business processes.

The development of collective clustering solutions for Big Data analysis pertaining to network traffic issues in large companies is proposed using a Bayesian approach, which enables adaptive updating of algorithm weights based on new data.

The use of an exponential dependency for calculating weights, which will amplify the differentiation of algorithm weights, is proposed, particularly when differences in quality metrics are minor. This may potentially make the ensemble CA (collective decision) method more sensitive to varying levels of performance. The idea is based on the premise that small changes in quality metric values result in more substantial changes in weights, which is important, because more efficient algorithms should have significantly greater weights due to their stronger impact on the final result. This is achieved by employing a Bayesian approach to update prior probabilities based on new data, thereby increasing the model's accuracy and adaptability. A co-associative matrix of differences, incorporating the weights of the base algorithms, enables the consideration of the reliability of clustering results depending on the specific algorithm used.

We propose enhancements to the workflow of the ensemble of clustering algorithms using various distance metrics for company network traffic analysis. Applying

hierarchical clustering for the final stage enables the derivation of final clusters that incorporate the results of all algorithms within the ensemble. For the Big Data analysis task in a large company's network, individual small groups of traffic can be aggregated into larger categories. For instance, specific applications may be grouped into broader usage categories, such as 'social networks' or 'business applications.' Hierarchical clustering also does not require a predefined number of clusters. This is suitable for Big Data, where the number of clusters is unknown.

The stages of an ensemble of clustering algorithms employing various distance metrics for company network traffic analysis are formulated. The proposed solutions further develop established ensemble (collective) cluster analysis approaches by introducing enhancements, including the construction of a co-association dissimilarity matrix taking into account the weights of base algorithms, the use of an exponential function to intensify differences in weights, consideration of the probabilistic relationship between ensemble performance characteristics and clustering quality indicators, and the application of a Bayesian approach to update prior probabilities based on new data. These innovations enable more accurate and reliable analysis and processing of Big Data network traffic in large enterprises.

Despite the positive results obtained, the proposed approach has a number of limitations. Experiments have shown that constructing the co-association matrix remains computationally expensive with extremely large traffic volumes. Further optimization of distributed implementations is required. Furthermore, the current version of the algorithm is oriented towards batch processing. At this stage, it does not account for the specifics of real-time stream analysis. Currently, the method has not been tested on encrypted traffic. Aadaptation to conditions with limited feature availability is planned for the next stage of the research. We consider integration of the ensemble with anomaly detection systems (IDS/IPS) to be a promising direction for further development. In addition, the application of hybrid schemes involving deep learning to improve clustering quality is foreseen.

References

[1] K. Takyi, A. Bagga, and P. Goopta, "Clustering techniques for traffic classification: A comprehensive review," in 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), IEEE, Aug. 2018, pp. 224-230. doi: 10.1109/ICRITO.2018.8748772.

[2] J. Li, H. Zhang, D. Tang, and C. Lin, "Traffic classification using cluster analysis," in 2021 International Conference on Computer Information Science and Artificial Intelligence (CISAI), IEEE, Sep. 2021, pp. 463-467. doi: 10.1109/CISAI54367.2021.00094.

[3] J. E. Rodriguez Rodriguez, V. H. M. Garcia, and M. A. O. Usaquén, "Corporate networks traffic analysis for knowledge management based on random interactions clustering algorithm" in Knowledge Management in Organizations: 13th International Conference, KMO 2018, Žilina, Slovakia, Aug. 2018, pp. 523-536. https://doi.org/10.1007/978-3-319-95204-8_44

[4] J. MacQueen, "Some methods for classification and analysis of multivariate observations" in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, no. 14, pp. 281-297, 1967.

[5] S. Lloyd, "Least squares quantization in PCM" in IEEE Transactions on Information Theory, vol. 28, no. 2, pp. 129-137, March 1982, doi: 10.1109/TIT.1982.1056489.

[6] Zhang, W., Zhang, L., Zhang, X., Wang, Y., Liu, P., & Gui, G. (2023). Intelligent Unsupervised Network Traffic Classification Method Using Adversarial Training and Deep Clustering for Secure Internet of Things. Future Internet, 15(9), 298.

[7] Zhang, T., Ramakrishnan, R. & Livny, M. BIRCH: A New Data Clustering Algorithm and Its Applications. Data Mining and Knowledge Discovery 1, 141–182 (1997). https://doi.org/10.1023/A:1009783824328

[8] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. 1998. CURE: an efficient clustering algorithm for large databases. SIGMOD Rec. 27, 2 (June 1998), 73–84. https://doi.org/10.1145/276305.276312

[9] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96). AAAI Press, 226–231.

[10] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. OPTICS: ordering points to identify the clustering structure. SIGMOD Rec. 28, 2 (June 1999), 49–60. https://doi.org/10.1145/304181.304187

[11] K. Subramani, A. Velkov, I. Ntoutsi, P. Kroger and H. -P. Kriegel, "Density-based community detection in social networks," 2011 IEEE 5th International Conference on Internet Multimedia Systems Architecture and Application, Bangalore, India, 2011, pp. 1-8, doi: 10.1109/IMSAA.2011.6156357.

[12] S. Zander, T. Nguyen and G. Armitage, "Automated traffic classification and application identification using machine learning," The IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05)l, Sydney, NSW, Australia, 2005, pp. 250-257, doi: 10.1109/LCN.2005.35.

[13] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow clustering using machine learning techniques" in Passive and Active Network Measurement: 5th International Workshop, PAM 2004, Springer, Apr. 2004, pp. 205-214. doi: 10.1007/978-3-540-24668-8_21

[14] Peter Cheeseman and John Stutz. 1996. Bayesian classification (AutoClass): theory and results. Advances in knowledge discovery and data mining.

American Association for Artificial Intelligence, USA, 153–180.

[15] Aouedi, O., Piamrat, K., Hamma, S. et al. Network traffic analysis using machine learning: an unsupervised approach to understand and slice your network. Ann. Telecommun. 77, 297–309 (2022). https://doi.org/10.1007/s12243-021-00889-1

[16] Pedro Casas, Alessandro D'Alconzo, Tanja Zseby, and Marco Mellia. 2016. Big-DAMA: Big Data Analytics for Network Traffic Monitoring and Analysis. In Proceedings of the 2016 workshop on Fostering Latin-American Research in Data Communication Networks (LANCOMM '16). Association for Computing Machinery, New York, NY, USA, 1–3. https://doi.org/10.1145/2940116.2940117

[17] Seydali, M., Khunjush, F. & Dogani, J. Streaming traffic classification: a hybrid deep learning and big data approach. Cluster Comput 27, 5165–5193 (2024). https://doi.org/10.1007/s10586-023-04234-0

[18] M. Hirvonen and J. -P. Laulajainen, "Two-phased network traffic classification method for quality of service management," 2009 IEEE 13th International Symposium on Consumer Electronics, Kyoto, Japan, 2009, pp. 962-966, doi: 10.1109/ISCE.2009.5157009.

[19] A. K. Jain, "Data clustering: 50 years beyond K-means" Pattern Recognition Letters, vol. 31, no. 8, pp. 651-666, 2010. https://doi.org/10.1016/j.patrec.2009.09.011

[20] Ghosh, J. and Acharya, A. (2011), Cluster ensembles. WIREs Data Mining Knowl Discov, 1: 305-315. https://doi.org/10.1002/widm.32

[21] Berikov, V. (2011). A Latent Variable Pairwise Classification Model of a Clustering Ensemble. In: Sansone, C., Kittler, J., Roli, F. (eds) Multiple Classifier Systems. MCS 2011. Lecture Notes in Computer Science, vol 6713. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-21557-5_30.

[22] K. Djouzi and K. Beghdad-Bey, "A Review of Clustering Algorithms for Big Data," 2019 International Conference on Networking and Advanced Systems (ICNAS), Annaba, Algeria, 2019, pp. 1-6, doi: 10.1109/ICNAS.2019.8807822.

[23] Shirkhorshidi, A.S., Aghabozorgi, S., Wah, T.Y., Herawan, T. (2014). Big Data Clustering: A Review. In: Murgante, B., et al. Computational Science and Its Applications – ICCSA 2014. ICCSA 2014. Lecture Notes in Computer Science, vol 8583. Springer, Cham. https://doi.org/10.1007/978-3-319-09156-3_49.

[24] Saeed MM, Al Aghbari Z, Alsharidah M. 2020. Big data clustering techniques based on Spark: a literature review. PeerJ Computer Science 6:e321 https://doi.org/10.7717/peerj-cs.321.

[25] Gao, M.; Ma, L.; Liu, H.; Zhang, Z.; Ning, Z.; Xu, J. Malicious Network Traffic Detection Based on Deep Neural Networks and Association Analysis. Sensors 2020, 20, 1452. https://doi.org/10.3390/s20051452.

[26] Berikov, V.B. Construction of an optimal collective decision in cluster analysis on the basis of an averaged co-association matrix and cluster validity indices. Pattern Recognit. Image Anal. 27, 153–165 (2017). https://doi.org/10.1134/S1054661816040040.

[27] Bian, Z., Qu, J., Zhou, J., Jiang, Z., & Wang, S. (2024). Weighted adaptively ensemble clustering method based on fuzzy co-association matrix. Information Fusion, 103, 102099. https://doi.org/10.1016/j.inffus.2023.102099

[28] 28. Zhu, Z., Xu, M., Ke, J., Yang, H., & Chen, X. M. (2023). A Bayesian clustering ensemble Gaussian process model for network-wide traffic flow clustering and prediction. Transportation Research Part C: Emerging Technologies, 148, 104032. https://doi.org/10.1016/j.trc.2023.104032

[29] Huang, Y., Jiang, H., & Ching, W. K. (2024). scEWE: high-order element-wise weighted ensemble clustering for heterogeneity analysis of single-cell RNA-sequencing data. Briefings in Bioinformatics, 25(3). https://doi.org/10.1093/bib/bbae203

[30] Wei, X., Zhang, Z., Huang, H., & Zhou, Y. (2024). An overview on deep clustering. Neurocomputing, 590, 127761. https://doi.org/10.1016/j.neucom.2024.127761

[31] Cherukuri, A. K., Ikram, S. T., Li, G., & Liu, X. (2024). Encrypted Network Traffic Analysis. In Encrypted Network Traffic Analysis (pp. 19-45). Cham: Springer International Publishing.

[32] Krajewska, A., & Niewiadomska-Szynkiewicz, E. (2024). Clustering Network Traffic Using Semi-Supervised Learning. Electronics, 13(14), 2769. https://doi.org/10.3390/electronics13142769

[33] Zhou, S., Duan, R., Chen, Z., & Song, W. (2024). Weighted ensemble clustering with multivariate randomness and random walk strategy. Applied Soft Computing, 150, 111015. https://doi.org/10.1016/j.asoc.2023.111015

[34] Aouedi, O., Piamrat, K., & Parrein, B. (2022). Ensemble-based deep learning model for network traffic classification. IEEE Transactions on Network and Service Management, 19(4), 4124-4135. doi: 10.1109/TNSM.2022.3193748

[35] Canadian Institute for Cybersecurity. (2017). CICIDS 2017 dataset. Canadian Institute for Cybersecurity. https://www.unb.ca/cic/datasets/malmem-2021.html

[36] 36. Moustafa, N., & Slay, J. (2015). The UNSW-NB15 dataset for network intrusion detection systems. Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), 1-7. https://doi.org/10.1109/MilCIS.2015.7348937 https://www.unsw.edu.au/about-us/our-story/our-story/cybersecurity

[37] Garcia, S., & Zetter, K. (2019). CTU-13 dataset: A botnet traffic capture dataset. CTU University. Retrieved from https://www.stratosphereips.org/datasets/ctu-13-dataset