International Journal of Communication Networks and Information Security
ISSN: 2073-607X, 2076-0930
Volume 15 Issue 04 Year 2023

Clustering Methods for Network Data Analysis in Programming

Vitaliy Kurdyukov
Senior Lecturer, Department of Physics and IT, M.Kh. Dulaty Taraz Regional University, Taraz, Republic of Kazakhstan
vitaliy_kurd@outlook.com
Galim Kaziev
Professor, Department of Information Systems and Cybersecurity, Almaty University of Power Engineering and Telecommunications named after Gumarbek Daukeev, Almaty, Republic of Kazakhstan
galim_kaziev@hotmail.com
Konysbek Tattibekov*
Associate Professor, Department of Higher Mathematics and Physics, Almaty Technological University, Almaty, Republic of Kazakhstan
konysbek.tattibekov@gmail.com

Article History

Received: 1 November 2023
Revised: 18 November 2023
Accepted: 30 December 2023

Abstract

In the modern world, data volumes are constantly increasing, and clustering has become an essential tool for identifying patterns and regularities in large datasets. The relevance of this study is associated with the growing need for effective data analysis methods in programming. The objective is to evaluate different clustering techniques within the programming domain and explore their suitability for analysing a wide range of datasets. Inductive and deductive methodologies, concrete illustrations, and visual techniques were employed. The clustering techniques were implemented using RStudio and Matlab tools. The study's findings facilitated the identification of crucial attributes of clustering techniques, including hierarchical structure, cluster quantity, and similarity metrics. The application of several data analysis and visualisation approaches, including k-means, c-means, hierarchical, least spanning tree, and linked component extraction, was illustrated. This study elucidated the clustering approaches that may be optimally employed in various contexts, resulting in enhanced precision in analyses and data-informed decision-making. The study's practical significance is in enhancing programmers' methodological toolset with tools for data analysis and processing.

Keywords: Data Mining, Programming Algorithms, Object Grouping, Cluster Analysis, Information Arrays

1. Introduction

Today, in an information society, the amount of data is consistently increasing, and the problem of handling this data has become crucial for different domains, including programming. Mastery of contemporary clustering techniques is a fundamental need in the programming domain. With the advancement of information technology and the growth of data quantities, the need to create and optimise clustering algorithms for diverse purposes such as customer segmentation, social network

analysis, image processing, and anomaly detection in data remains significant. The significance of the study stems from the need for efficient analysis and processing of various types of data. Although clustering techniques are widely used, there remain significant concerns and challenges that warrant further investigation. These include selecting the most suitable clustering technique for a specific task, determining the optimal number of clusters, and addressing the unique characteristics of different data types. Moreover, the rapid advancement in programming technology and the introduction of new data types call for continuous updates and modifications to clustering algorithms.

Researchers in various countries, including Kazakhstan, have been exploring issues related to data clustering within the realm of programming. One study examined methods for clustering vast amounts of data to address real-world programming challenges. The main goal was to develop programs in programming languages that meet specific optimality criteria and generate data clusters. Another research focused on data processing techniques in the medical field, analyzing infections and patients' medical records to create models, establish data processing algorithms, and use clustering techniques for data characterization, employing the k-means algorithm and density-based clustering with AUTO configuration. A different approach proposed a methodology for identifying and automatically resolving discrepancies in large datasets used for urban transportation analysis, incorporating a versatile module that operates on two levels, using statistical methods and machine learning algorithms, including the k-means clustering method and a multilayer neural network, to detect and rectify discrepancies. Additionally, a study investigated a clustering technique to manage co-reference relationships in the Kazakh language, aiming to consolidate personal names related to individuals in texts by using various methods, such as the Tomita-parser and clustering algorithms, to enhance the accuracy of extracting and combining named entities. Finally, a cluster analysis conducted on the regions of Kazakhstan using the "STATISTICA" computer system and hierarchical classification methods identified two main clusters of regions with similar socio-economic characteristics, reporting the results of dividing a specific set of areas into clusters using Ward's method and full linkage techniques on dendrograms.

The purpose of this study is to review contemporary clustering methods in programming and investigate their applicability for analyzing specific datasets. The following tasks are addressed: selecting the optimal clustering method for programming, determining the optimal number of clusters, adapting methods to new data types and evaluating the effectiveness of methods in practical tasks.

2. Related Works

[6] introduced two methods for determining the number of centers in fuzzy clustering based on two-level programming. The first method employs an evolutionary algorithm and has been practically validated, while the second method combines mean shift and fuzzy clustering. The results of numerical experiments show that both proposed methods successfully determine the number of cluster centers and enhance data analysis and image segmentation. [7] investigated the problem of modelling time-varying operations in complex optimization tasks for energy supply systems. They proposed using clustering methods, including traditional ones (k-means, hierarchical clustering) and shape-based methods. The authors compared the effectiveness of these methods using examples of battery optimization and gas turbine planning. The study revealed that centroid-based clustering methods more reliably represent operational aspects of optimization tasks but are biased in evaluating the objective function.

In his research, [8] considered the problem of effective clustering in wireless sensor networks (BSN) within the Internet of Things (IoT). The study offered a novel approach that integrates the merits of the k-medoids method with the affinity propagation (AP) method. Initially, the affinity propagation (AP) technique is employed to ascertain the count of cluster heads and choose the most suitable cluster centres for k-medoids. Then, the modified k-medoids method is utilized to construct

the network topology. This approach allows for a more uniform distribution of cluster heads, enhancing network performance [9, 10].

Authors in [11] introduce an online clustering method called Contrastive Clustering (CC), which performs contrastive learning at both the instance and cluster levels. Positive and negative instance pairs were created in the dataset, and their representations in the feature space were optimized for contrastive learning. This method simultaneously improves cluster representations and assignments, allowing for cluster assignments to be computed for new data [12, 13]. In the context of classification tools, [14] presented a comparison of Python and R programming languages, highlighting the functionality of their packages. The aim of the study is to identify changes in the Mariana Trench data. The methodology included hierarchical cluster analysis and the construction of cluster maps with marginal dendrograms. The results revealed three distinct groups of profiles grouped by height ranges with maximum depths. Dendrogram visualization in cluster analysis effectively represents data grouping, sorting, and classification using machine learning algorithms [15]. The presented software codes allow sorting datasets in similar studies to group data based on attribute similarities [16-18].

The paper by [19] proposes a clustering method based on the analysis of social networks with experimental and control groups in a programming course for freshmen. The study identified a significant improvement in learning efficiency, and female students in the experimental group demonstrated better social outcomes [20]. A new clustering method was proposed by [21]. It considers various size constraints, iteratively minimizes errors by assigning data to clusters based on size constraints, optimizes integer linear programming, and updates cluster centers [22].

Authors in [23] address the problem of task allocation among a set of robots with balance constraints to improve their efficiency. They introduced a balance constraint that minimizes travel differences between robots and ensures an equal number of tasks for each robot. To solve this problem, they utilized clustering methods, analyzed clustering approaches, and their applicability considering balance criteria using a dataset. The results showed that the k-means clustering method is most suitable for solving problems with complex topologies and can scale to work with different task and robot quantities compared to other clustering methods [24], [25].

In the study by [26], specific open-source software for clustering is presented, including open-source code. Clustering was implemented in programming languages such as R, Java, C++, and Python. In the paper by [27], user behaviors in smart card systems are classified in the context of public transportation demand analysis. As classical methods are unsuitable for working with time series, a classification method for users' daily smart card transactions based on cross-correlation measure, hierarchical clustering, and metric parameters was proposed [28], [29]. The clustering results were compared with the dynamic time warping method. The authors also developed a programme in R and tested the method on smart card transaction data from a public transportation system. The results showed that cross-correlation is more effective for classifying time series, and this method can identify different user behavior patterns.

Clustering plays a crucial role in data analysis and machine learning, involving the grouping of similar data points according to specific criteria or patterns. A range of clustering algorithms exists to address diverse applications and challenges across numerous industries (see Table 1).

Table 1. Summary of Clustering Methods

Method	Features	Advantages	Limitations
Fuzzy clustering methods	Evolutionary algorithm	Successfully determines cluster centers	Limited information on other methods used
	Mean shift and fuzzy clustering	Enhances data analysis	Specific to fuzzy clustering
Clustering for energy supply systems	Traditional and shape-based methods	Effective for modeling time-varying ops	Centroid-based methods are biased
		Comparatively evaluates optimization	
Clustering in wireless sensor networks	Combines k-medoids and AP methods	Uniform distribution of cluster heads	Complex method description
		Enhances network	

Method	Features	Advantages	Limitations
		Performance	
Contrastive clustering	Contrastive learning	Simultaneously improves clusters	Limited information on dataset used
	Positive and negative instance pairs	Allows cluster assignments for new data	
Comparison of Python and R	Hierarchical cluster analysis	Effective data grouping and sorting	Limited to Python and R comparison
	Dendrogram visualization	Utilizes machine learning algorithms	
Social network analysis	Experimental and control groups	Improved learning efficiency	Oversimplified assumptions about network homogeneity
	Better social outcomes (female)		Scale sensitivity and data sparsity
Size-constrained clustering	Size constraints and optimization	Minimizes errors with size constraints	Requires integer linear programming
	Iterative data assignment	Updates cluster centers	
Task allocation for robots	Balance constraint	Minimizes travel differences	Focuses on balance constraints
	K-means clustering	Suitable for complex topologies	
Open-source software for clustering	Multiple programming languages	Access to open-source software	Lack of comprehensive documentation and user support
User behavior classification	Cross-correlation measure	Effective for classifying time series	Specific to time series data
	Hierarchical clustering	Identifies user behavior patterns	

3. Methodology

This study combined inductive and deductive reasoning with real-world examples and visuals to explore how data can be clustered in programming. Through inductive methods, the research uncovered patterns and structures within the data that weren't apparent before clustering. Induction involves drawing broader conclusions from specific examples, which in this case, meant examining individual data points to find commonalities and emerging patterns. This approach enabled the discovery of new relationships within the data, enhancing understanding of its core characteristics and significant elements. By clustering similar data points, distinct patterns and traits became evident, providing deeper insights into the data's fundamental qualities.

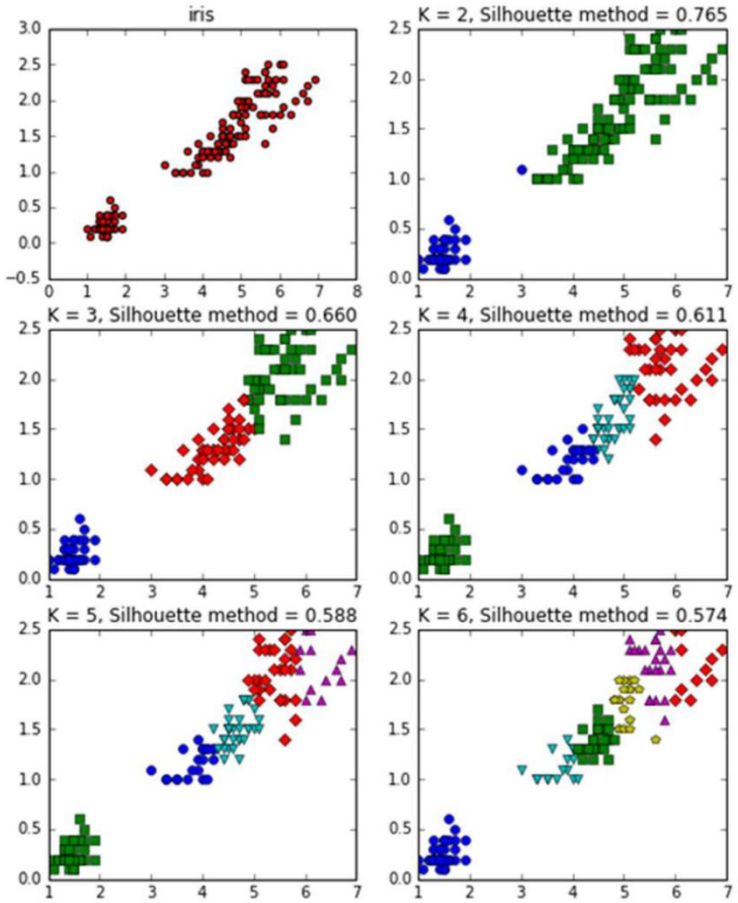

Figure 1. Research on Value Selection Method of Clustering Algorithm

This research used deductive methods to identify patterns in the data and come to conclusions. Deductive reasoning involves using logic to draw specific conclusions from general statements. The study carefully built upon already spotted patterns to see what further conclusions could logically follow. Combining deductive with inductive methods, and supporting these with real examples and visuals, allowed for a thorough investigation into data clustering in programming. Inductive methods helped discover underlying patterns, while deductive reasoning was applied to interpret these patterns, thereby enriching the study's findings.

In practical examples, clustering methods such as k-means, c-means, hierarchical clustering, minimum spanning tree, and connected component extraction were examined. These methods were implemented using various simple programs developed using R and Matlab languages. Each programme corresponded to a specific clustering algorithm and performed its task. The results of practical examples not only validated the theoretical effectiveness of the algorithms but also served as a basis for formulating clear recommendations for their application in the field of programming. Visual methods played a crucial role in this study, allowing the results of the developed programmes to be visually represented.

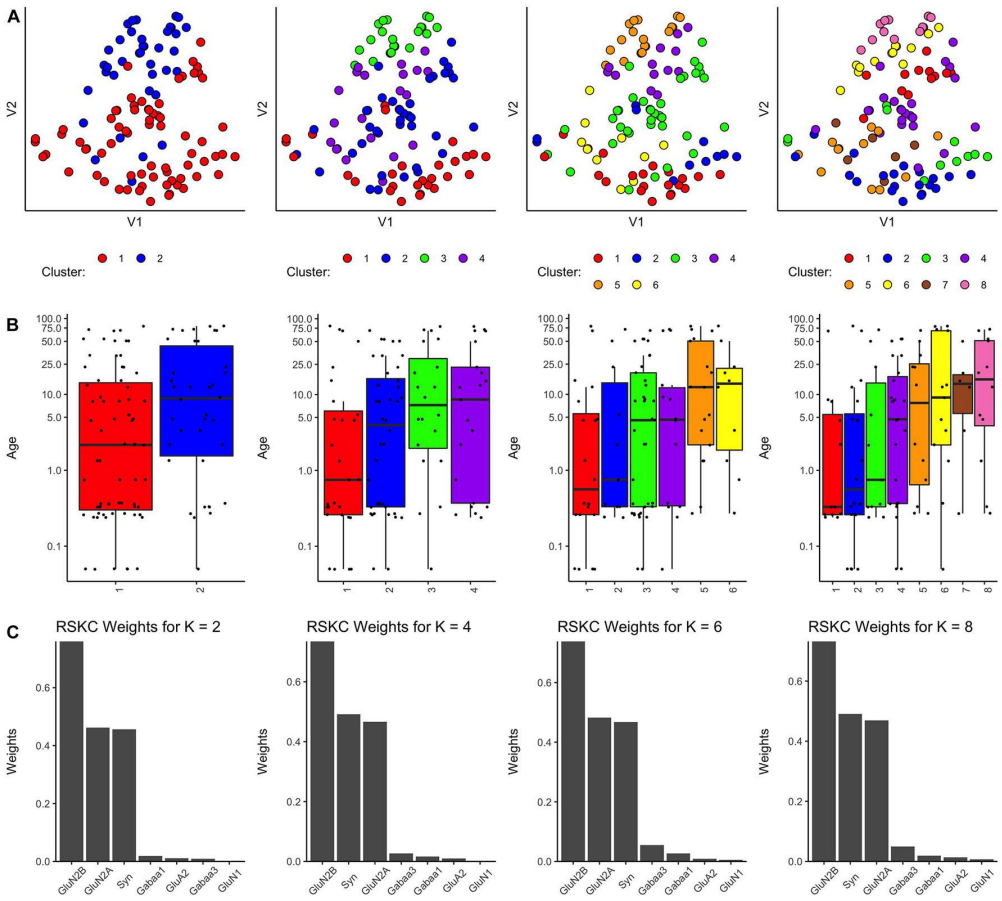

Figure 2. Practical Guide to Sparse Clustering

Standard scientific documents describing various methods, algorithms, and principles of clustering in the computer industry were studied, and small programmes were written to implement certain clustering algorithms using RStudio and Matlab tools. Specific formulas were used to assess the computational complexity when comparing clustering algorithms. For k-means and c-means algorithms: $O(nkl)$, for the hierarchical method: $O(n^2)$, for the minimum spanning tree: $O(n^2 \log n)$, and for hierarchical clustering: $O(\max(n, m)), m < n(n-1)/2$. Thus, the combination of methods used allowed for in-depth exploration in the field of data clustering in programming.

4. Results and Discussion

Clustering is an essential tool for data analysis with numerous applications in the modern world. Utilising clustering methods in the field of computer science allows for the automation of data analysis processes and pattern recognition. This, in turn, can lead to more efficient and informative decision-making in various domains. In general, clustering is the process of grouping similar objects or data into clusters, where objects within the same cluster are similar to each other, and objects from different clusters are dissimilar. Various classifications of clustering algorithms can be identified: based on data processing methods (hierarchical and flat algorithms) and based on data analysis methods (crisp and fuzzy algorithms). Hierarchical algorithms (taxonomy algorithms) construct a system of nested partitions of objects into clusters, resulting in a cluster tree where the root represents the entire dataset, and the leaves represent individual objects (smallest clusters). Flat algorithms create a single partition of objects in the dataset into non-overlapping clusters. In crisp (non-overlapping) algorithms, each object in the dataset is assigned a cluster number, meaning each object belongs to only one cluster. In fuzzy (overlapping) algorithms, each object is associated with a set of real values indicating the degree of membership to clusters, meaning each object belongs to each cluster with a certain probability.

During the iterations of cluster analysis, various linkage methods are applied, which serve as criteria during the merging (for agglomerative algorithms) or splitting (for divisive algorithms) of clusters. Among the linkage methods, determining the proximity between two clusters, individual linkage methods are identified, including the following: single linkage, complete linkage, and group average methods. In the single linkage method, the distance between clusters is determined by the distance between the closest objects in these clusters, often leading to clusters tending to form chains. In the complete linkage method, the distance between clusters is determined by the distance between the farthest objects in these clusters. It is used when clusters appear as widely separated groups of points. The method of inter-group linkage can be unweighted or weighted. In unweighted linkage, the distance between clusters is calculated as the average distance between all pairs of objects in the clusters. This method is effective when objects form distinct groups and works well for long (chain-like) clusters. In weighted linkage, the distance between clusters is calculated differently from the unweighted method, taking into account weight coefficients, which represent the sizes of the clusters (the number of objects in them). This method is applied when clusters have unequal sizes.

The next category of linkage methods involves methods based on the linkage between cluster centers. When these methods are used, distances between the centers of weighted clusters, also known as centroids, are calculated after adding an object to a cluster. This category includes centroid-based methods, median-based methods, and dispersion-based methods. Centroid-based methods can be unweighted or weighted. In unweighted centroid linkage, the distance between clusters is determined by the distance between the centroids of the weighted clusters. At each stage of the algorithm, the centroid of each cluster is located at the point with the average coordinates of all objects in the cluster. Weighted centroid linkage differs from the unweighted method by taking into account the weight coefficients, representing the sizes of the clusters (the number of objects in them). In median-based methods, distances between any cluster and the new cluster formed by combining two clusters are determined as the distance between the cluster and the midpoint of the line segment connecting the merged clusters. The Ward's method, a dispersion-based method, is based on a different logic. It combines not the clusters that are maximally close in some sense but those whose merger results in the smallest increase in within-cluster dispersion, minimizing the "spreading" of clusters formed at previous stages.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm that groups data points based on their density, effectively identifying clusters of varying shapes and sizes. In turn, Mean Shift is a mode-seeking algorithm that iteratively shifts data points towards areas of higher data density, converging to cluster centroids [30]. Hierarchical Agglomerative Clustering (HAC) builds a hierarchy of clusters by recursively merging or splitting clusters based on a defined linkage criterion. In addition, Gaussian Mixture Model (GMM) represents data as a mixture of Gaussian distributions and uses the Expectation-Maximization algorithm to estimate the parameters of these distributions. Self-Organizing Maps (SOM) are unsupervised neural networks that map high-dimensional data onto a lower-dimensional grid, preserving the data's topological relationships [31].

Various features can be highlighted to classify different algorithms. For instance, the method of object allocation to clusters includes bottom-up and top-down algorithms. In bottom-up algorithms, objects are initially grouped into a single cluster and then successively divided into smaller clusters. In top-down algorithms, each object is initially assigned its own cluster, and these clusters are sequentially merged until the desired level of separation is achieved. Quadratic error algorithms create clusters based on a mathematical expression to compute the root mean square deviation. Artificial intelligence systems employ neural networks for object separation. In the logical approach, data is divided into clusters using decision trees. Despite all types, approaches and methods of clustering, it is customary to distinguish the following algorithms: k-means, c-means, hierarchical, allocation of connected components, minimal spanning tree and layered clustering. There are other algorithms (methods) as well; however, these are the most popular and widely used ones (Table 2).

Table 2. Comparison of Clustering Algorithms

Clustering algorithm	**Cluster shape**	**Computati-onal**	**Input data**	**Results**

		complexity		
K-means	Hypersphere	$O(nkl)$	Number of clusters	Cluster centers
C-means			Number of clusters and degree of fuzziness	Cluster centers and membership matrix
Hierarchical	Arbitrary	$O(n^2)$	Number of clusters or distance threshold for hierarchy truncation	Binary cluster tree
Selection of connected components		Depends on the algorithm	Distance threshold R	Tree structure of clusters
Minimum spanning tree		$O(n^2 \log n)$	Number of clusters or distance threshold for edge removal	
Layered clustering		$O(\max(n, m) < n(n - 1)/2$	Sequence of distance thresholds	Tree structure of clusters with different levels of hierarchy

Note: k - number of clusters, l - number of iterations.

The work of the aforementioned clustering algorithms can be implemented in RStudio by writing code in the R programming language. RStudio is an integrated development environment (IDE) that provides a user interface and a set of tools for working with R, a programming language and environment for statistical analysis and data visualization. RStudio can be used not only for data analysis but also for descriptive statistics, hypothesis testing, creating graphs, and regression analysis [32]. It is recommended to use it for the following reasons: interactive data analysis (RStudio provides an interactive environment for data analysis, allowing quick data loading, visualization, and pre-processing before applying clustering methods); rich library (R has an extensive package library, including many tools for clustering data); integration with visualization (RStudio integrates data visualization directly into the development environment); community and documentation (R and RStudio have active user communities and extensive documentation, including numerous tutorials and books on data analysis and clustering); support for other languages (RStudio can also work with other programming languages, such as Python, expanding its functionality and enabling the use of various clustering methods available in different languages). In summary, this platform offers a comfortable and robust environment for programming, data analysis, and the implementation of clustering algorithms.

The code sample in RStudio using the R language to implement the k-means clustering algorithm:

```
# Installation and loading of the library
install.packages("stats")
library(stats)
# Creating random data for clustering
set.seed(123) # for reproducibility
data <- data.frame(
x = rnorm(100, mean = 0, sd = 1),
y = rnorm(100, mean = 3, sd = 1)
)
# Selecting the number of clusters
k <- 3
# Application of the K-means algorithm
```

```
kmeans_result <- kmeans(data, centers = k)
# Output information about the clusters
print(kmeans_result)
# Visualisation of clustering results
plot(data, col = kmeans_result$cluster)
points(kmeans_result$centers, col = 1:k, pch = 8, cex = 2)
```

The code creates random data, applies a k-means algorithm with three clusters, and outputs information about them. It also builds a graph to visualize the results (Figure 3).

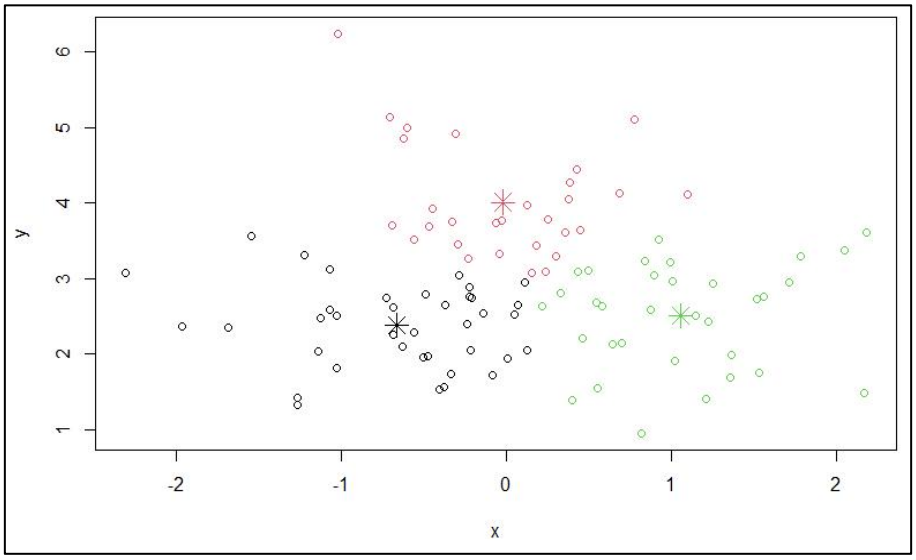

Figure 3. The Result of the K-means Algorithm

An example of code for hierarchical clustering can also be provided:

```
# Loading data for example
data(USArrests)
# Performing hierarchical clustering
hc <- hclust(dist(USArrests), method = “ward.D2”)
# Drawing a dendrogram (cluster tree)
plot(hc, cex = 0.6, hang = -1)
```

The code performs three main actions: First, it loads the "USArrests" dataset, which includes arrest statistics for various US states over a specific period. Then, it applies the "hclust" function to perform hierarchical clustering on this dataset. Lastly, it generates a dendrogram, a type of diagram that illustrates the arrangement of the clusters formed, which can be seen in Figure 4.

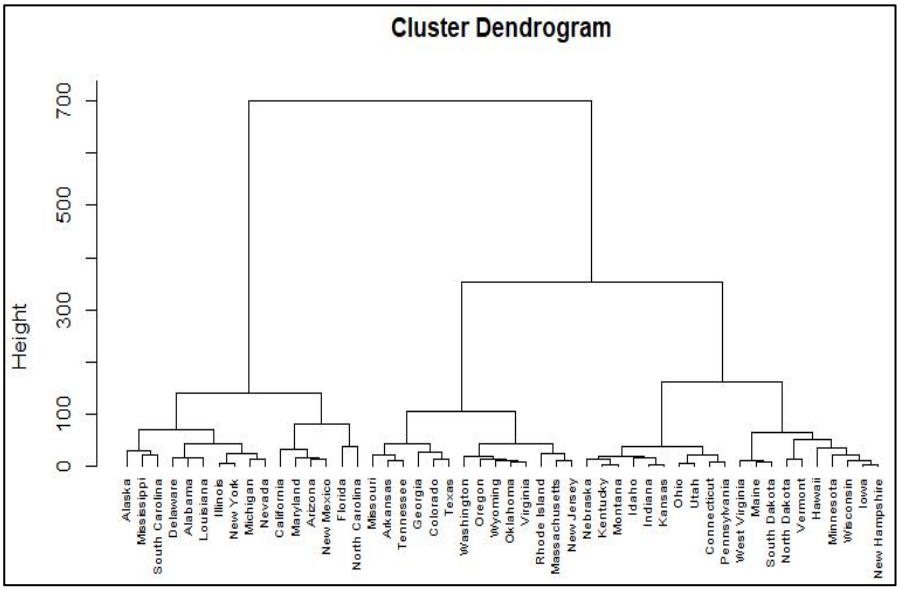

Figure 4. The Result of Hierarchical Clustering

Matlab is a versatile software that works well with RStudio for a variety of clustering tasks. It serves as both an advanced programming language and an interactive platform, mainly used for performing numerical calculations, visualizing data, developing and analyzing algorithms, creating models, among other functions. While Matlab is favored in the software industry for its capabilities in handling numerical computations, its installation process and high memory requirements can pose challenges. The software comes equipped with a broad range of data manipulation tools, including those for clustering, facilitating straightforward visualization of results. Moreover, Matlab's support for parallel computing allows for the efficient handling of large datasets, a critical aspect in clustering operations.

Matlab code snippet demonstrating the implementation of the c-means clustering algorithm:

```
% Generating random data for example
rng('default'); % Setting the initial value for the random number generator
data = [randn(100,2)+1.5; randn(100,2)-1.5];
% Selection of the number of clusters
k = 2;
% Initialisation of random centroids
centroids = datasample(data, k, 'Replace', false);
% Maximum number of iterations
maxIterations = 100;
tolerance = 1e-4;
for iter = 1:maxIterations
% Finding the nearest centroid for each point
distances = pdist2(data, centroids);
[~, clusterIndices] = min(distances, [], 2);
% Centroid update
newCentroids = zeros(k, size(data, 2));
for i = 1:k
newCentroids(i, :) = mean(data(clusterIndices == i, :));
end
% Convergence check
if max(abs(newCentroids(:) - centroids(:))) < tolerance
break;
```

```
end
centroids = newCentroids;
end
% Visualisation of results
figure;
gscatter(data(:,1), data(:,2), clusterIndices);
hold on;
plot(centroids(:,1), centroids(:,2), 'kx', 'MarkerSize', 10, 'LineWidth', 2);
title('Clustering results using c-means algorithm (C-Means)');
legend('Cluster 1', 'Cluster 2', 'Centroids', 'Location', 'Best');
hold off;
```

This code implements the c-means clustering algorithm on random data. It starts by generating random two-dimensional data, sets the number of clusters for separation, and then iterates, finding the nearest clusters for each data point and updating centroids. The results are visualized on a plot, where each cluster is represented by a different color, and centroids are indicated by black crosses (Figure 5).

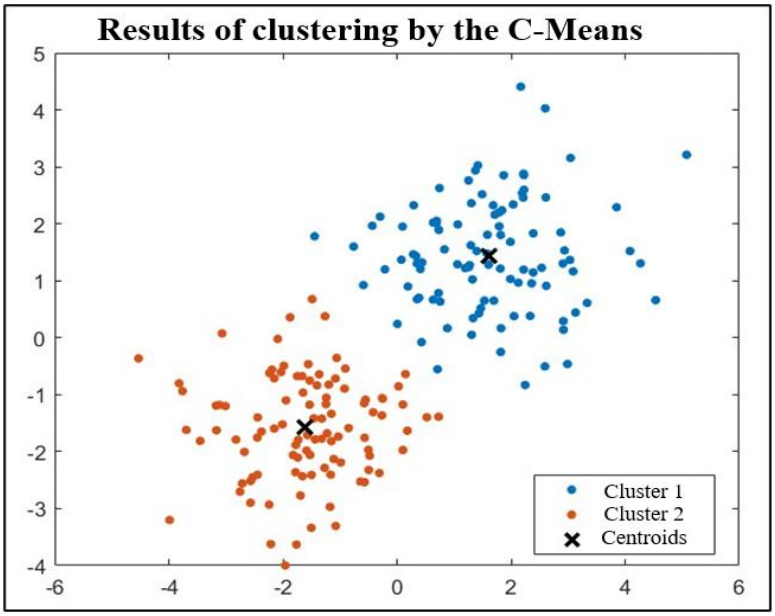

Figure 5. The Result of the C-means Algorithm

The clustering algorithm with a minimum spanning tree can also be implemented:

```
% Random data generation
rng('default'); % Setting the initial value for the random number generator
data = [randn(100, 2); randn(100, 2) + 3];
% Calculating pairwise distances between the data points
distances = pdist2(data, data);
% Creating a graph based on distances
G = graph(distances);
% Obtaining the minimum spanning tree
T = minspantree(G);
% Visualising the edges of the minimum spanning tree
figure;
plot(T, 'EdgeLabel', T.Edges.Weight);
title('Minimum Spanning Tree');
```

```
% Setting the number of clusters
k = 2;
% Extraction of clusters from the edges
clusters = conncomp(T);
% Visualisation of clustering results
figure;
gscatter(data(:, 1), data(:, 2), clusters);
title('Clustering Using Minimum Spanning Tree');
```

The programme starts by generating random data, calculating pairwise distances between points, creating a graph based on these distances, and then finding the minimum spanning tree in this graph. The clustering results are visualized, and the points are divided into clusters represented by different colors on the plot (Figure 6).

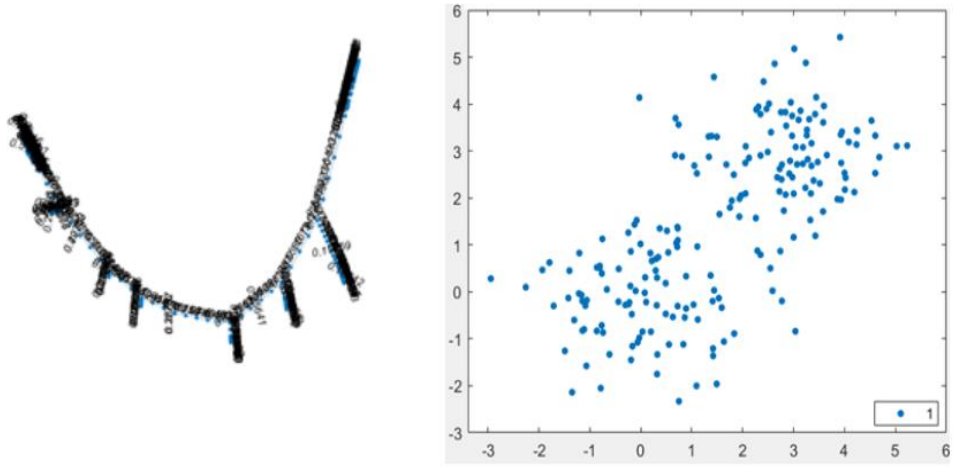

Figure 6. The Result of the Minimum Spanning Tree Algorithm

It is also possible to implement clustering algorithms in programming for various tasks. For example, there is a graph that is represented as a list of edges. The task is to find all connected components in this graph (groups of vertices in the graph where each vertex is connected to another vertex within the same group but not connected to vertices in different groups). The goal is to find and output all such groups of vertices. This problem involves the algorithm for finding connected components. The solution in the form of R code would be as follows:

```
# Graph represented as an edge list
edges <- data.frame(
from = c(1, 2, 3, 4, 5, 6, 7, 8),
to = c(2, 3, 1, 4, 5, 6, 7, 8)
)
# Installing the igraph library
install.packages("igraph")
library(igraph)
# Creating a graph from the edge list
graph <- graph_from_data_frame(edges, directed = FALSE)
# Finding connected components
components <- clusters(graph)
# Output of connected components
for (i in 1:length(components$csize)) {
cat(paste("Connected component", i, ": ", components$csize[i], "vertices(s)\n"))
cat(paste("Vertices:", toString(components$membership[components$membership== i]), "\n\n"))
}
```

This code creates a graph from the edge list and uses the clusters function from the igraph library to find connected components in the graph. Then, it outputs information about each connected component, including the list of vertices belonging to that component (Table 3).

Table 3. The Result of the Connected Components Extraction Algorithm

Connected component 1 Vertices: 1, 1, 1	3 vertices
Connected component 2 Vertices: 2	1 vertex
Connected component 3 Vertices: 3	1 vertex
Connected component 4 Vertices: 4	1 vertex
Connected component 5 Vertices: 5	1 vertex
Connected component 6 Vertices: 6	1 vertex

Also, the authors requested other clustering algorithms. Thus, for DBSCAN, set the epsilon (neighborhood distance) and minimum points parameters.

```
from sklearn.cluster import DBSCAN
from sklearn.datasets import make_moons
import matplotlib.pyplot as plt
# Generate synthetic data
X, _ = make_moons(n_samples=200, noise=0.05, random_state=0)
# DBSCAN clustering
dbscan = DBSCAN(eps=0.3, min_samples=5)
labels = dbscan.fit_predict(X)
# Plot the results
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis')
plt.title("DBSCAN Clustering")
plt.show()
For Mean Shift, adjust the bandwidth parameter.
from sklearn.cluster import MeanShift
from sklearn.datasets import make_blobs
import matplotlib.pyplot as plt
# Generate synthetic data
X, _ = make_blobs(n_samples=300, centers=4, cluster_std=0.6, random_state=0)
# Mean Shift clustering
meanshift = MeanShift()
labels = meanshift.fit_predict(X)
# Plot the results
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis')
plt.title("Mean Shift Clustering")
plt.show()
```

For HAC, choose the linkage criterion and set the number of clusters.

```
from sklearn.cluster import AgglomerativeClustering
from sklearn.datasets import make_blobs
import matplotlib.pyplot as plt
# Generate synthetic data
```

```
X, _ = make_blobs(n_samples=300, centers=4, cluster_std=0.6, random_state=0)
# HAC clustering
hac = AgglomerativeClustering(n_clusters=4)
labels = hac.fit_predict(X)
# Plot the results
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis')
plt.title("Hierarchical Agglomerative Clustering")
plt.show()
```

For GMM, specify the number of components and initialize parameters.

```
from sklearn.mixture import GaussianMixture
from sklearn.datasets import make_blobs
import matplotlib.pyplot as plt
# Generate synthetic data
X, _ = make_blobs(n_samples=300, centers=4, cluster_std=0.6, random_state=0)
# GMM clustering
gmm = GaussianMixture(n_components=4)
labels = gmm.fit_predict(X)
# Plot the results
plt.scatter(X[:, 0], X[:, 1], c=labels, cmap='viridis')
plt.title("Gaussian Mixture Model Clustering")
plt.show()
```

For SOM, define the grid size and learning rate.

```
from minisom import MiniSom
import numpy as np
import matplotlib.pyplot as plt
# Generate random data
data = np.random.rand(100, 2)
# SOM training
som = MiniSom(10, 10, 2, sigma=1.0, learning_rate=0.5)
som.train_random(data, 100)
# Plot the SOM grid
plt.pcolor(som.distance_map().T, cmap='bone_r')
plt.colorbar()
plt.title("Self-Organizing Map")
plt.show()
```

[34] present a method for efficient Dynamic Time Warping (DTW) and clustering of time series data. The method considers DTW as an optimization problem solved using dynamic programming and then clusters time series data by solving a second optimization problem using mixed-integer programming (MIP). The authors also proposed an option to use k-medoids clustering to increase speed when a global optimality certificate is not required. This approach was tested using a time series archive and demonstrated a 33% speed increase compared to other clustering methods, which increases to 64% for larger datasets. [35] improved the fuzzy k-means clustering method, one of the simplest k-means clustering methods, by adding a recurrent neural network, leading to the creation of a new method. In this method, the error of fuzzy k-means clustering is modelled through a constrained optimization problem [36]. Simulation results on academic datasets confirmed the effectiveness of the proposed method [37]. The common aspect between these studies is their use of the k-means clustering algorithm.

[38] investigated the clustering of software modules, allowing for a better understanding of complex software systems by dividing them into more manageable modules. The authors concluded that efficient automatic methods for clustering software modules are needed to manage resources. In their paper, they conducted a literature review on clustering models and presented a taxonomy for classifying existing research, categorizing clustering methods into three main classes. In addition, they identified new challenges and areas for future research in the field of software module clustering. [39] examined distance metric learning methods in clustering that utilize information provided by experts in the form of constraints. They transform the data so as to comply with these restrictions. However, since this transformation can alter the data distribution, the authors proposed a method using Lagrange multipliers, which helps metric learning algorithms preserve the original data distribution while considering the provided constraints [40]. The results show that this method provides good clustering performance and a more accurate representation of data after transformation.

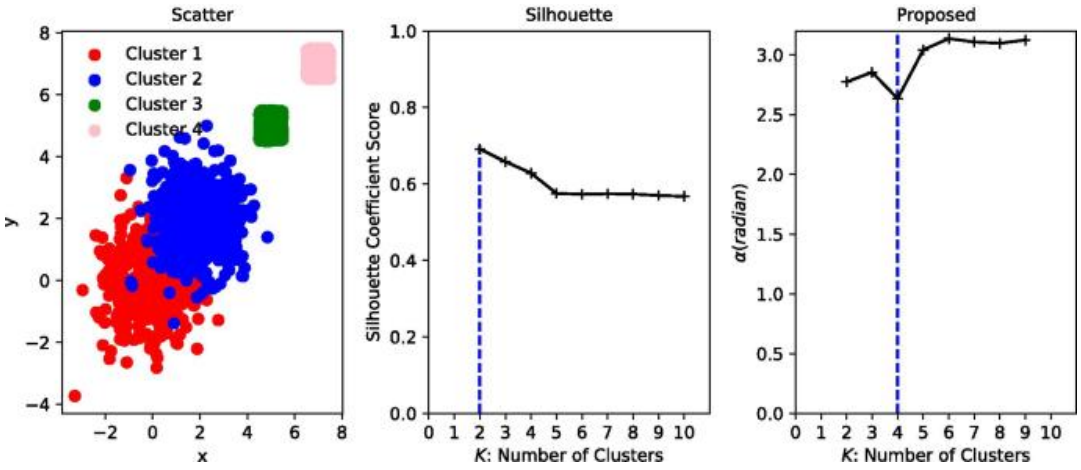

Figure 7. A Quantitative Discriminant Method of Elbow Point for the Optimal Number

[41] has illustrated that network communication datasets are essential for grasping and scrutinizing the dynamics of today's computer networks. These datasets include diverse details about network traffic, protocols, and communication trends. To deeply analyze these extensive datasets, understanding the various data kinds, their configurations, and volumes is necessary. Techniques such as fuzzy clustering, size-constrained clustering, and contrastive clustering play a crucial role in breaking down network communication data. These methods reveal hidden structures within the data, like communication trends, irregularities, and user behavior categories. In scenarios like energy supply systems or wireless sensor networks, clustering helps in resource distribution, load balancing, and assigning tasks efficiently. Using methods from social network analysis can also illuminate the connections within networks and identify significant nodes, contributing to network efficiency enhancement. The choice between Python and R for conducting clustering analysis hinges on the user's particular needs and their comfort level with these languages, as both provide strong open-source capabilities. By applying various clustering techniques, researchers can gain a richer insight into network behaviors, boost network efficiency, and enhance security surveillance in intricate communication settings.

Choosing the right clustering algorithm is critical to the success of data analysis projects. Table 4 outlines key considerations from the research findings that can assist in selecting the most fitting clustering algorithm for specific data and objectives.

Table 4. Criteria for Selecting Clustering Methods in Data Analysis

Criteria	Relevant Clustering Methods
Data characteristics	DBSCAN (for data with varying densities); Mean Shift (mode-seeking for data with density peaks); K-means (for well-separated, spherical clusters); Hierarchical Clustering (for hierarchical structures); Gaussian Mixture Model (for data with Gaussian-like distribution); Self-Organizing Maps

Criteria	Relevant Clustering Methods
	(for preserving data topology); Fuzzy Clustering (for assigning membership degrees)
Number of clusters	Methods based on validation metrics (e.g., silhouette score, elbow method); Evolutionary algorithms (for automatically determining the number of centers)
Cluster shape and size	DBSCAN (for arbitrary cluster shapes); Mean Shift (for flexible cluster shapes); K-means (may require careful initialization for non-spherical clusters); Hierarchical Clustering (flexible for various cluster shapes and sizes)
Interpretability of clusters	K-means (each point belongs to one cluster); Fuzzy Clustering (assigns membership degrees)
Efficiency and scalability	K-means (efficient for large datasets); DBSCAN (may struggle with high-dimensional data); Self-Organizing Maps (may require tuning for large datasets)
Handling noise and outliers	DBSCAN (identifies noise points as outliers); Mean Shift (may not handle noise well); Hierarchical Clustering (may need to prune small clusters as noise)
Real-time or batch processing	DBSCAN (can be adapted for real-time streaming data); K-means (suitable for batch processing)
Need for hierarchical clustering	Hierarchical Clustering (naturally supports hierarchy)
Ease of implementation and availability	K-means (widely available in libraries and tools); DBSCAN (available in popular libraries); Mean Shift (commonly implemented); Gaussian Mixture Model (available in libraries); Self-Organizing Maps (may require custom implementation); Fuzzy Clustering (available but less common)

Thus, K-means is suitable for tasks where data needs to be divided into a predefined number of clusters. C-means, utilizing fuzzy logic, is beneficial when data objects can belong to multiple clusters with varying degrees of certainty. Hierarchical clustering provides a cluster tree, which is useful for analyzing data at different levels of detail. Minimum spanning tree can be employed to find minimal connections between data objects. This is useful in optimizing network connections, route planning, and other tasks were finding the smallest costs for connecting points is necessary. Layered clustering is suitable for analyzing data with multiple levels of clustering, for instance, in organizing large hierarchical data networks.

5. Conclusion

In this study, various papers dedicated to clustering methods in the computer science field and programming were analyzed. The most popular clustering methods were implemented, and the results of the work of the programs were visualized using RStudio and Matlab tools. The purpose of the study included the utilization of specific clustering algorithms in the context of programming and the analysis of different types of data for their implementation. To achieve this purpose, methods were employed that enhanced the efficiency of clustering algorithms and facilitated the development of new methods capable of successfully addressing the set tasks. The strategies used in the study led to more precise results in data processing and object categorization. It became clear that there are many different clustering techniques, algorithms, and principles, each showing better performance for specific tasks.

Based on this study's findings, we can offer some recommendations. When selecting a clustering technique, it's essential to closely examine the specific task at hand since the effectiveness of various methods can differ greatly depending on the characteristics of the data and the objectives of the study. Utilizing a combination of methods and algorithms can lead to better clustering results, particularly for datasets with diverse attributes. For research in specific areas, integrating relevant domain-specific considerations when choosing and setting up clustering algorithms can make the results more understandable. Properly preparing the data beforehand and choosing the right ways to

measure distances between data points are crucial steps that can significantly impact the success of clustering efforts. Using visual tools like dendrograms to map out the clusters can greatly aid in understanding the data structure, helping to interpret the results and make informed decisions. Providing software versions of newly developed methods or algorithms can be very useful for other researchers. Furthermore, it's important to keep pushing forward with research in clustering, as new methods and types of data continue to emerge, aiming to create algorithms that are both more precise and efficient.

To sum up, clustering algorithms play a key role in the analysis and organization of data within programming. For every task, there is an ideal clustering approach that can yield the best possible outcomes. Persistent research into clustering techniques is critical to address the evolving landscape of data types and specific research needs.

References

[1] V. V. Kurdyukov, "Big data clustering models," in P*roceedings of the 1st International Student Conference "Modern Trends in Design, Construction, Repair and Maintenance of Transport Structures"*, Minsk: Belarusian National Technical University, 2017, pp. 146-154.

[2] A. D. Kubegenova, and K. T. Iskakov, "Aspect on intellectual analysis and application of methods in medicine using data mining technology," *Bulletin of Toraighyrov University. Energy Series,* vol. 1, pp. 184-195, 2021.

[3] D. Kassymova, A. Tashev, A. Akhmediyarova, Z. Umirbekova, L. Kuntuova, "Approach to detecting and eliminating inconsistencies in big data bases for the tasks of city passenger transport analytic," *The Bulletin of Kazakh Academy of Transport and Communications named after M. Tynyshpaev*, vol. 121, no. 2, pp. 600-611, 2022.

[4] G. Kalman, M. G. Esmaganbet, M. M. Zhamankarin, A. I. Gabdulina, D. V. Pleskachev, "Conference solution using the clustering method," *News of the Academy of Sciences of the Republic of Kazakhstan*, vol. 1, no. 345, pp. 121-135, 2023.

[5] A. Zh. Yesbulatova, "Clustering of regions by the real sector of the economy of the Republic of Kazakhstan," *Science and Education*, vol. 3, no. 2, p. 225-233, 2023.

[6] K. Qiao, J. Zhang, and J. Chen, "Two effective heuristic methods of determining the numbers of fuzzy clustering centers based on bilevel programming," *Applied Soft Computing,* vol. 132, p. 109718, Jan. 2023.

[7] H. Teichgraeber and A. R. Brandt, "Clustering methods to find representative periods for the optimization of energy systems: An initial framework and comparison," *Applied Energy,* vol. 239, pp. 1283-1293, Apr. 2019.

[8] J. Wang, Y. Gao, K. Wang, A. Sangaiah, and S.-J. Lim, "An Affinity Propagation-Based Self-Adaptive Clustering Method for Wireless Sensor Networks," *Sensors,* vol. 19, no. 11, p. 2579, Jun. 2019.

[9] M. Novitasari, Yaddarabullah, D. Handy, and Erneza Dewi Krishnasari, "Classification of house buildings based on land size using the K-nearest neighbor algorithm," Jan. 2022.

[10] H. Petryshyn, O. Kryvorychko, H. Lukashchuk, N. Danylko, and O. Klishch, "Changing the Qualities of Urban Space by Means of Landscape Architecture," *Architectural Studies,* vol. 8, no. 1, 2022.

[11] Y. Li, P. Hu, Z. Liu, D. Peng, J. T. Zhou, and X. Peng, "Contrastive Clustering," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 35, no. 10, pp. 8547-8555, May. 2021.

[12] H. Y. Suhendi, M. A. Ramdhani, and F. S. Irwansyah, "Board Governance Attributes and Organizational Characteristics of Mosque Co-Operatives in Malaysia," *International journal of engineering & technology,* vol. 7, no. 2.29, pp. 321-321, May. 2018.

[13] Manafaddin Namazov, Viacheslav Matsiuk, Iuliia Bulgakova, Iryna Nikolaienko, and P. B. Верниgора, "Agent-based simulation model of multimodal iron ore concentrate transportation," *Machinery & energetics,* vol. 14, no. 1, Feb. 2023.

[14] P. Lemenkova, "R Libraries {dendextend} and {magrittr} and Clustering Package scipy.cluster of Python For Modelling Diagrams of Dendrogram Trees," *Carpathian Journal of Electronic and Computer Engineering,* vol. 13, no. 1, pp. 5-12, Sep. 2020.

[15] V. Lysenko, I. Bolbot, A. I. MARTYNENKO, Taras Lendiel, and Kateryna Nakonechna, "PROGRAM IMPLEMENTATION OF MOBILE PHYTOMONITORING WORK," *Machinery & Energetics,* vol. 13, no. 1, Apr. 2022.

[16] M. Kutia, J. Li, A. J. Sarkissian, and T. Pagella, "Land cover classification and urbanization monitoring using Landsat data: A case study in Changsha city, Hunan province, China," *Ukrainian journal of forest and wood science,* vol. 14, no. 1, pp. 72-91, Feb. 2023.

[17] IURII SHYNKARIUK, "Alternative Representation of Space and Time: Geometric Solution of Problems of Relativity Theory," *Naukovij vìsnik Užgorods'kogo unìversitetu*, no. 51, Jun. 2022, doi: https://doi.org/10.54919/2415-8038.2022.51.74-82.

[18] D. Fernandez, Omkar Dastane, Hafizah Omar Zaki, and A. Aman, "Robotic process automation: bibliometric reflection and future opportunities," May. 2023, doi: https://doi.org/10.1108/ejim-10-2022-0570.

[19] W. C. Chang, Y. J. Fan, and A. R. Chang, "Cooperative learning clustering in the programming courses," in *Frontier Computing: Theory, Technologies and Applications (FC 2022)*, Singapore: Springer, 2023, pp. 52-59.

[20] S. M. Kenesbayev, G. I. Salgarayeva, A. A. Makhmetova, S. N. Idrissov, and B. Sabit, "Management of information software systems in the corrective work with children with disabilities," *Management,* vol. 38, no. 46, p. 34, 2017.

[21] W. P. Tang, Y. Yang, L. Zeng, and Y. Zhan, "Size Constrained Clustering With MILP Formulation," *IEEE Access,* vol. 8, pp. 1587-1599, Jan. 2020.

[22] Almaz Tlemisovich Mustafin and Kazakh National Research Technical University named after K.I. Satpayev, "SYNCHRONOUS OSCILLATIONS OF POPULATIONS OF TWO SPECIES LINKED BY DIRECT COMPETITION," *Izvestiya Vysshikh Uchebnykh Zavedeniy*, vol. 23, no. 4, pp. 3-23, Jan. 2015, doi: https://doi.org/10.18500/0869-6632-2015-23-4-3-23.

[23] E. Murugappan, N. Subramanian, S. Rahman, M. Goh, and H. K. Chan, "Performance analysis of clustering methods for balanced multi-robot task allocations," *International Journal of Production Research,* vol. 60, no. 14, pp. 4576-4591, Aug. 2022.

[24] R. Zadorozhniuk, "UAV data collection parameters impact on accuracy of Scots pine stand mensuration," *Ukrainian journal of forest and wood science,* vol. 14, no. 1, Feb. 2023.

[25] V. P. Babak et al., "Models of measuring signals and fields," *Models and Measures in Measurements and Monitoring,* pp. 33-59, 2021, doi: https://doi.org/10.1007/978-3-030-70783-5_2.

[26] D. J. Hand, "Data Clustering: Theory, Algorithms, and Applications by Guojun Gan, Chaoqun Ma, Jianhong Wu," *International Statistical Review*, vol. 76, no. 1, p. 141, Apr. 2008, doi: https://doi.org/10.1111/j.1751-5823.2007.00039_2.x.

[27] L. He, B. Agard, and M. Trépanier, "A classification of public transit users with smart card da ta based on time series distance metrics and a hierarchical clustering method," *Transportmetri ca A: Transport Science*, pp. 1-20, Jun. 2018, doi: https://doi.org/10.1080/23249935.2018.1479722.

[28] Yaddarabullah, M. F. Muttaqin, and M. Rafiansyah, "Service-Oriented Architecture for E-Marketplace Model Based on Multi-Platform Distributed System," *IOP Conference Series: Materials Science and Engineering,* vol. 662, p. 042028, Nov. 2019.

[29] A. Tanchak, K. Katovský, I. Haysak, J. Adam, and R. Holomb, "Research of spallation reaction on plutonium target irradiated by protons with energy of 660 MeV," *Naukovij vìsnik Užgorods'kogo unìversitetu,* no. 52, Nov. 2022, doi: https://doi.org/10.54919/2415-8038.2022.52.36-45.

[30] A. Fahim, "A varied density-based clustering algorithm," *Journal of Computational Science,* vol. 66, p. 101925, Jan. 2023.

[31] N. Shirani-bidabadi, R. Ma, and M. W. Anderson, "Within-day travel speed pattern unsupervised classification - A data driven case study of the State of Alabama during the COVID-19 pandemic," *Journal of Traffic and Transportation Engineering (English Editing)*, vol. 8, no. 2, pp. 170-185, Apr. 2021.

[32] F. Kronthaler, S. Zöllner, and Springerlink, "Online Service," *Data Analysis with RStudio : An Easygoing Introduction.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2021.

[33] L. Yu, "Matlab Programming Environment Based on Web," *IEEE Xplore,* Dec. 01, 2018, doi:https://ieeexplore.ieee.org/document/8740716.

[34] V. Kumtepeli, R. Perriment, and D. A. Howey, "Fast dynamic time warping and clustering in C++," *arXiv (Cornell University),* Jul. 2023, doi: https://doi.org/10.48550/arxiv.2307.04904.

[35] Karim El Moutaouakil and A. Touhafi, "A New Recurrent Neural Network Fuzzy Mean Squa re Clustering Method," Nov. 2020, doi: https://doi.org/10.1109/cloudtech49835.2020.9365873.

[36] A. Mustafin, "Awakened Oscillations in Coupled Consumer-Resource Pairs," *Journal of Applied Mathematics,* vol. 2014, pp. 1-20, Jan. 2014.

[37] F. F. Arwy, Yaddarabullah, and H. Permana, "Clustering Ornamental Plants Turnover Data using K-Means Algorithm," in *2021 4th International Conference of Computer and Informatics Engineering (IC2IE),* Sep. 2021.

[38] J. Li and A. Yamini, "Clustering-based software modularisation models for resource management in enterprise systems," *Enterprise Information Systems,* pp. 1-21, Oct. 2020.

[39] R. Randel, D. Aloise, and A. Hertz, "A Lagrangian-based approach to learn distance metrics for clustering with minimal data transformation," in *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, Philadelphia: SIAM, 2023, pp. 127-135.

[40] T. C. Chen et al., "Application of Data Mining Methods in Grouping Agricultural Product Customers," *Mathematical Problems in Engineering,* vol. 2022, pp. 1-9, Mar. 2022.

[41] B. Mou and X. Yang, "Analysis of the Role of Compliance Plan in AI Criminal Risk Prevention-Take AI Criminal Risk in Network Communication as Example," *International Journal of Communication Networks and Information Security (IJCNIS)*, vol. 15, no. 3, pp. 154-167, Dec. 2023.