A Hybrid Clustering Approach for link prediction in Heterogeneous Information Networks

Zahra Sadat Sajjadi
Islamic Azad University
Mahdi Esmaeili
Islamic Azad University
Mostafa Ghobaei-Arani (mo.ghobaei@iau.ac.ir)
Islamic Azad University
Behrouz Minaei-Bidgoli
Iran University of Science and Technology

Research Article

Keywords: Social Network, Graph Clustering, Structural Similarity, Attribute Similarity, Hybrid Similarity, K-Medoids

Posted Date: March 2nd, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2626833/v1

Additional Declarations: No competing interests reported.

A Hybrid Clustering Approach for link prediction in Heterogeneous Information Networks

Zahra Sadat Sajjadi [1], Mahdi Esmaeili [2], Mostafa Ghobaei-Arani [*1], Behrouz Minaei-Bidgoli [3]

[1] Department of Computer Engineering, Qom Branch, Islamic Azad University, Qom, Iran

[2]Department of Computer Engineering, Kashan Branch, Islamic Azad University, Kashan, Iran

[3]School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

*Corresponding author's email: mo.ghobaei@iau.ac.ir

Abstract:

In recent years, researchers from academic and industrial fields have become increasingly interested in social network data to extract meaningful information. This information is used in applications such as link prediction between people groups, community detection, protein module identification, etc. Therefore, the clustering technique has emerged as a solution to finding similarities between social network members. Recently, in most graph clustering solutions, the structural similarity of nodes is combined with their attribute similarity. The results of these solutions indicate that the graph's topological structure is more important. Since most social networks are sparse, these solutions often suffer from insufficient use of node features. This paper proposes a hybrid clustering approach for link prediction in heterogeneous information networks (HINs). In our approach, an adjacency vector is determined for each node until, in this vector, the weight of the direct edge or the weight of the shortest communication path among every pair of nodes is considered. A similarity metric is presented that calculates similarity using the direct edge weight between two nodes and the correlation between their adjacency vectors. Finally, we evaluated the effectiveness of our proposed method using DBLP and Political blogs datasets under entropy, density, purity, and execution time metrics. The simulation results demonstrate that while maintaining the cluster density significantly reduces the entropy and the execution time compared with the other methods.

Keywords: *Social Network, Graph Clustering, Structural Similarity, Attribute Similarity, Hybrid Similarity, K- Medoids*

1. Introduction

Nowadays, social networks are very popular for facilitating and modelling the communication between different social groups [1]. These social networks provide a place for exchanging opinions and sharing people's views and feelings. Social networks contain vast and valuable data, and helpful information can be obtained from analyzing these data. Networks are divided into homogeneous and heterogeneous. In a homogeneous network, all objects and connections between them are of the same type. A heterogeneous network consists of nodes, which represent different types of objects, and edges, which establish relationships between them. In a social network, information can be shown with heterogeneous information networks (HINs). Various networks, including computer networks, social networks, signalling networks and etc., are usually modelled by graphs as an effective tool for examining objects and their relationships. The objects are associated with different attributes to enrich the information content of a network. Graph clustering is an exciting and challenging research field due to the difficult structures and connections between objects in the real world. As a result, various aspects of graph clustering have been studied to gain a better understanding of network structure and semantics [2]. The effective factor in clustering is finding a similarity criterion between objects so that the criterion is consistent with the purpose of clustering [3]. The similarity between objects is calculated according to their topological structure or feature. The state-of-the-art methods use only one of the two aspects. The S-Cluster algorithm is a baseline clustering algorithm that only considers topological structure [4,7]. The other baseline algorithm K-SNAP partitions a graph such that each partition has nodes with identical attribute values [5]. In other words, the similarity of objects is measured based on only one of two aspects. In these methods, clustering is not quality because much of the network information is ignored during the similarity calculation and the clustering process.

Using the combined similarity measure effectively solves this limitation [2,6-18]. However, in clustering the objects of a network into different clusters based on the combination of two aspects, the structural relationships are still more effective than the characteristics of the nodes. For example, most of these methods cannot use the property of nodes completely. Therefore, the extracted clusters may be inaccurate, especially when the network is sparse.

The purpose of this paper is to perform the clustering process on HINs with considering attributes. The proposed solution uses the graph clustering solution considering structure and context to achieve desired quality at a lower computational cost. It takes into account the type of connection

between nodes. Then, it calculates the adjacency vector for each node based on its relationships with other nodes and provides a similarity measure using the Pearson correlation coefficient. After that, the k-Medoids algorithm is applied to cluster the nodes based on their similarity score.
The contributions of this work are summarized as follows:

- We proposed a hybrid clustering approach for heterogeneous information networks, which uses the k-Medoids technique to partition nodes based on the combined similarity value.
- We use the importance of disconnected nodes' presence to calculate the similarity between nodes.
- We perform experiments on DBLP and Political blogs datasets regarding density, entropy, and purity metrics to evaluate our solution.

The remaining parts of this paper are organized as follows: Section 2 examines related work on graph clustering for link prediction in social networks. In section 3, we explain the proposed solution in more detail. Section 4 provides an evaluation of the proposed method and discusses the results. Finally, we present conclusions and future research to develop the current work in Section 5.

2. Related works

This section will discuss the different approaches for graph clustering and link prediction problems using structure and attribute similarities in complex networks. Besides, we summarize the research studies to solve the graph clustering and link prediction problem.
Ghorbanzadeh et al. [19] have proposed a new method for solving the link prediction problem using common neighbourhoods in directed graphs. Their proposed method used the authority, hub, and neighbourhood direction. Their solution performs in both supervised and unsupervised models. Further, they evaluated their strategy on the SmaGri, Wiki-vote, Political blogs, and Kohonen real-world datasets. They illustrated that their method outperforms in terms of precision and sensitivity metrics than with other methods.
Zarei et al. [20] have proposed an approach for solving link prediction using hidden relations among users in social networks. Their proposed method categorizes each node's neighbours to calculate the similarity score between a pair of nodes. They used nine real-world datasets and demonstrated that their method was more accurate than the other methods.

In [21], the authors presented a link prediction approach for HINs via a deep convolutional neural network. The proposed method in link prediction based on community detection is performed in 4 steps: local neighborhood discovery, Local subgraph tensorization, Embedded learning, and link prediction. This approach was evaluated on four different types of HINs. In addition to applying to many scenarios, this approach has a reasonable execution time and can be used for various tasks.

According to [22], the solution is proposed to rank and predict links in a network such that it expands the random walks via a distinct restart probability for each node. The results on two datasets reveal that the proposed method outperforms the classic random walk with restart (RWR) regarding link prediction.

The label propagation algorithm for solving graph clustering has been improved by Berahmand et al. [23]. Their proposed version produced a weighted graph that is created from the initial graph by considering the node attributes and topological structure. Further, they evaluated their method on real and artificial datasets. They indicated that their approach is more efficient and precise on the criteria density, entropy, and Normalized Mutual information (NMI) index.

Agrawal et al. [24] have studied graph clustering for detecting communities that combine both topological and attribute similarities in terms of communication type to provide an efficient plan. Further, their proposed plan balances the distance function and executes clustering using k-Medoid background. They used datasets of DBLP and Political blogs and measured density, entropy, and NMI measures to demonstrate the effectiveness of their algorithm.

In [25], a strategy is proposed to solve link prediction in complex networks. The suggested technique uses path properties of different lengths to compute the similarity score between pairs of nodes. Their strategy has used the concept of allocation of network resources. This technique increases the quantity of information received at the destination node by limiting information leakage by shared neighbors and maximizes the two nodes' similarity score. This work has been tested on various datasets and evaluates this strategy against two measures AUC curve and average precision. The evaluation results revealed that their strategy differs considerably from the baseline techniques.

Kumar et al. [26] have introduced a new method to predict links based on level-2 node clustering coefficients. Their method presents level-2 common nodes and clustering coefficients to gather information about clusters from the seed node pair's level-2 familiar neighbors. They used eleven real-world datasets in their work and evaluated their method with the baseline methods in metrics ROC curve, AUPR curve, precision, and recall. In comparison with state-of-the-art algorithms, their proposed method showed superiority.

Ghasemi et al. [27] have proposed a clustering-based method to improve link prediction. Their method is done in two steps: The first step is offline and is executed once. This step calculates local and global metrics for each node using the available data. Then, the classification algorithm is used to develop the classification-based link prediction model. Algorithm Ada Boost has been used as the best classifier. A clustering technique is employed in step two to group social items using estimated similarity criteria. Furthermore, they tested their method on the Facebook, HepTh, and Brightkite datasets and evaluated that based on precision, recall, and fitness metrics.

Dmytro et al. [28] have presented a solution to predict links between objects in HINs. The HINs are analyzed to extract a meta-path, then links below a certain threshold level are removed, and their algorithm is used to calculate the connectional power. They used the Web of Science datasets to demonstrate their method's effectiveness.

According to Table 1, we reviewed and summarized graph clustering and link prediction approaches and compared them in terms of datasets, techniques used, and performance metrics.

Table 1. A comparison of the different graph clustering and link prediction approaches.

Reference	**technique used**	**Evaluating Tool**	**Performance Metric**	**Dataset**	**Disadvantage**	**Advantage**
(Ghorbanzadeh et al.,[19])	Hybrid-based	Simulation (Python)	Precision, Sensitivity	SmaGri, Wiki-vote, Political blogs, Kohonen	No forecast for the direction of the links	the best performance in unsupervised mode, Low computational complexity
(Zareie et al., [20])	Similarity-based	Simulation (Java)	Identifying connections between nodes without common neighbors, Identifying the relationships among nodes by the number of common neighbors, Accuracy	Nine different real-world networks	No regard for directed and weighted networks	superior accuracy results

(Xi Wang et al., [21])	Deep Embedding	Simulation (PyTorch)	F1- score, Computational efficiency(cost) , Accuracy, Precision, Recall, AUC	Wordnet, MovieLens, Douban, DBLP	No prediction for the direction of the links, Loss the information, No regard for multimedia contents	Better performance, the acceptable computational cost
(Woojeong Jin et al., [22])	Random Walk With Extended Restart	Simulation (MATLAB)	Accuracy, Speed, Scalability, Memory Usage, AUC	HepTh, HepPh	working on homogeneous and without weights graphs	The usage of a different restart probability for each node and the automated determination of restart probabilities
(Kamal Berahmand et al., [23])	Label Propagation (graph clustering)	Simulation (Python)	Density, Entropy, NMI, F1-Score	Cora, Citeseer, Political blogs, LFR-EA	The remove entropy from the results due to sparsity of the attributes of nodes	Linear time complexity, Suitable for large datasets
(Agrawal et al., [24])	Graph clustering	Simulation (JDK, Python)	Density, Entropy, NMI, Accuracy	DBLP, Political blogs	The quadratic time complexity for medium size graph	Combine both topological structure and attributes similarities
(Ajay Kumar et al., [25])	Path-based approach	Simulation (MATLAB)	AUROC, Average Precision	Networks such as Collaboration, Social, Citation, Biological	considering high-order path index lead to affect little bit to prediction accuracy	considering high-order path index lead to affect little bit to prediction accuracy, Reduce information leakage
(Ajay Kumar et al., [26])	Level-2 node clustering coefficient	Simulation (MATLAB)	AUROC, Accuracy, AUPR, average Precision, Recall	11 real-world datasets	No regard for directed and weighted networks, Poor predictive power compared to Nod2vec, SPM algorithms	Define the notion of the level-2 common node
(Ghasemi et al., [27])	Graph Clustering (Hybrid-based)	WEKA Project, AdaBoost Classifier	Precision, Recall, Fitness	Facebook, Brightkite, HepTh	High computational time due to global parameters	With local and global parameters, precision is higher than with baseline methods
(Dmytro et al., [28])	Meta-Path, Random Walk	VOSviewer software, Perl language	Number of restored links, Restored links percentage	Scientific collaboration networks (Web of Science)	High computational time, The impact of the sparseness of data on the predictive results	The use heterogeneous information network

3. Proposed Approach

In this section, an explanation of the proposed approach is described. First, a framework based on the combination of nodes' structural characteristics and attributes is presented. The clustering problem is then formulated. Finally, the proposed algorithm for graph clustering of heterogeneous information networks is explained.

3.1. Proposed Framework

This section will discuss a framework for combining topological structure and attribute of nodes to implement the suggested approach. As shown in Figure 1, the proposed framework includes five main steps: the data pre-processing, the connection extraction, the similarity calculation, the combining structural and attribute similarities, and the performing the process of clustering and evaluating the clusters, each of which is explained in the following:

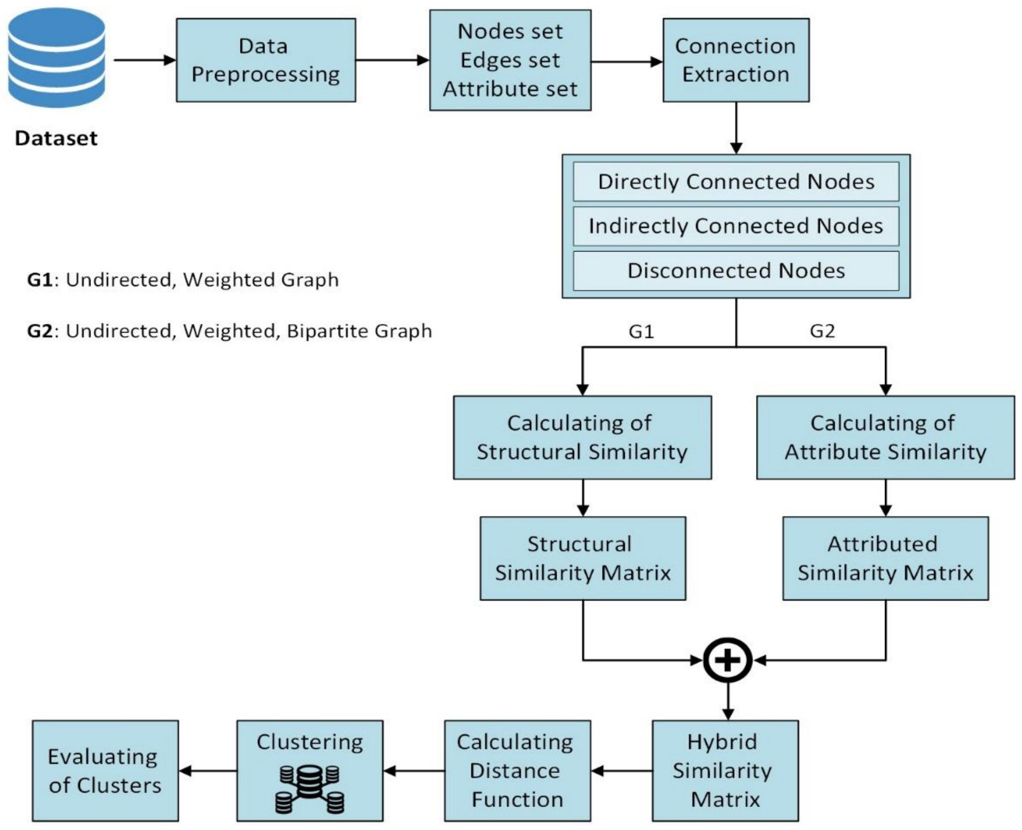

Dataset
Data Preprocessing
Nodes set
Edges set
Attribute set
Connection Extraction
Directly Connected Nodes
Indirectly Connected Nodes
Disconnected Nodes
G1: Undirected, Weighted Graph
G2: Undirected, Weighted, Bipartite Graph
G1
G2
Calculating of Structural Similarity
Calculating of Attribute Similarity
Structural Similarity Matrix
Attributed Similarity Matrix
Hybrid Similarity Matrix
Calculating Distance Function
Clustering
Evaluating of Clusters

Fig. 1. The proposed clustering algorithm framework

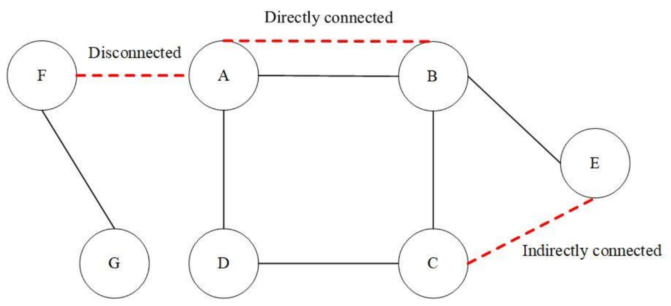
A. Data pre-processing

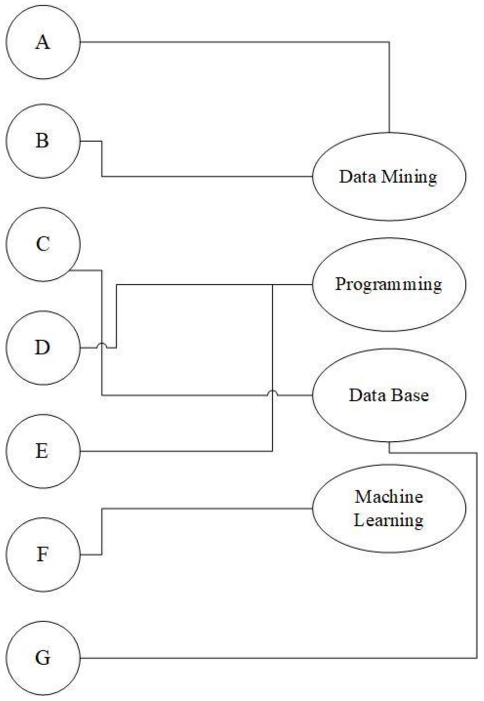
This step is responsible for pre-processing the input dataset. This step is divided into two processes filtering and coding. Filtering is in charge of data cleaning on the input dataset, and coding is responsible for building relationships between records within the data. The pre-processing step is carried out once, and its results are used in all other steps.

B. Connection extraction

The connection between node pairs is extracted once, and these connections are used in various steps. The nodes' connection is divided into three types: *Directly connected*, *Indirectly connected*, and *Disconnected*. *Directly connected*, in this connection type, there is a direct edge between two nodes. For example, in Figure 2-a, nodes *A* and *B* or *A* and *D* are *Directly connected*. *Indirectly connected*, in this kind of connection, there is no direct edge between two nodes, but a communication path passing through other nodes may establish a connection between two nodes. In Figure 2-a, nodes *E* and *C* are *Indirectly connected*. *Disconnected*, in this connection type, exists not a direct edge or a path between nodes. In the proposed method, these nodes may communicate with other nodes in the network based on common features.

In Figure 2-a, the connection between nodes *A* and *F* is called *Disconnected*. After extracting of connections, the data are modelled in the form of two graphs: the simple graph (*G1*) and the bipartite graph (*G2*).





(a) (b)

Fig. 2. Types of graphs a) Simple graph (*G1*), the dashed line shows the connection types; b) Bipartite graph (*G2*)

C. Similarity calculation

In this step, the structural similarity between node pairs in *G1* is calculated according to the type of connection between them separately. The result of this step is the structural similarity matrix. In addition, attribute similarity between node pairs in *G2* is calculated based on the type of connection between them separately. The result is an attribute similarity matrix.

D. Combining similarities

The hybrid similarity consists of the combined structural and attribute similarities between pairs of nodes according to their connection type. In this combination, the structural similarity is based only on the edges or communication paths between the nodes, and the attribute similarity is based only on node features. The output of this step is called the hybrid similarity matrix according to the influence degree of the two similarities.

E. Clustering and evaluating

In implementing the proposed algorithm, the K-Medoids method uses distance values for vertices partitioning. The outcome of the clustering is k clusters, each of which contains a set of vertices. Clusters are mutually separated and collectively complete. After the clustering process, the clusters will be evaluated using three criteria: density, entropy, and purity.

3.2. Problem statement

As shown in Table 2, this section introduces the notations and equations used in the proposed solution. The dataset is an undirected, weighted, multi-attributed graph $G = \{V, E, W, M, A\}$, not necessarily connected, where V and E are the set of all the vertices and undirected edges respectively, W is the weight of each edge, M is the number of node attributes, and A is the set of values of each attribute $A = \{attr_1, attr_2, \dots, attr_M\}$. Two graphs, *G1* and *G2,* are extracted from graph *G*. *G1* is an undirected and weighted graph $G1 = \{V1, E1, W1\}$, not necessarily connected, where *V1*, *E1*, and *W1* are adapted from *G*. If there is a direct link among any pairs of vertices, e.g., V_n and V_m, then $W1_{nm}>0$. Also, *G2* is an undirected, weighted bipartite graph $G2 = \{V2, E2, W2, M, A\}$. Each attribute appears as a single node in the bipartite graph. Therefore, *V2* equals the sum of *G1* nodes and each of the attributes these nodes have. The $E2$

attribute edge is an edge between a node and each of the characteristics of that node. $W2$ represents the edge weight for each attribute; by default, its value is equal to one. Also, M and A are adapted from G. In $G1$ and $G2$, parameter d_n , indicates the degree of each node and the number of edges entered into it. In $G1$, CN_{nm} , is the number of common neighbors of two nodes, e.g., V_n and V_m. In $G2$, CN_{nm} , is the number of common attributes between two nodes. The goal is to partition the graph into k segments using the combination of topological and attribute similarities such that the nodes in a partition have strong structural relationships and homogeneous attribute values.

Table 2. Notations and definitions.

Notation	Definition
G	An undirected, weighted, multi-attributed graph
V	Set of vertices
E	Set of edges
M	Number of attributes of each node
A	Set of values of each attribute
W_{nm}	Edge weight between the vertices V_n and V_m
W_{atti}	i^{th} attribute edge weight
$V_n \leftrightarrow V_m$	Two vertices, V_n and V_m, are *Directly connected*
$V_n \not\leftrightarrow V_m$	Two vertices, V_n and V_m, are *Indirectly connected*
$V_n \cdots V_m$	Two vertices, V_n and V_m, are Dis*connected*
α	Impact parameter
$sim(V_n, V_m)_{Struct}$	Structural similarity between two vertices V_n, V_m
$sim(V_n, V_m)_{attribute}$	Attribute similarity between two vertices V_n, V_m
$HSim(V_n, V_m)$	The hybrid similarity between two vertices V_n, V_m
d_n	node degree
CN_{nm}	Common neighbors between two vertices V_n and V_m
$Distance(V_n, V_m)$	Distance function
K	Number of clusters
$corr_{nm}$	The correlation coefficient between two vertices, V_n and V_m

3.3. Proposed graph clustering algorithm

This section provides a detailed explanation of the clustering algorithm, as shown in Algorithm 1. Initially, the dataset must be processed before other steps can use it. The pre-processing consists of two processes: filtering and coding. In the filtering process, it is tried to extract a coherent dataset with a smaller volume than the initial dataset by applying appropriate filters. In the coding process, data coding is done with a simple coding method for greater integrity. The output of the pre-processing phase is the three sets of nodes, the edges, and the attributes of nodes. In the proposed algorithm, once data pre-processing (line 3) and extracting the connections type between vertices with each other (line 4) is performed. Then, a simple, undirected, and weighted graph (G1) is extracted as a model to solve the structural similarity problem. Also, a bipartite, undirected, and weighted graph (G2) is a model for solving the attribute similarity problem. According to the output of line 4, structural similarity and attribute (lines 6-17) are repeated for both vertices. Then, the hybrid similarity and distance function of each pair of vertices will be calculated (lines 17-22) and finally, will be done clustering process (line 23).

Algorithm 1: Pseudocode Proposed Clustering Algorithm

```
1:  Input: A Dataset, K: number of Clusters, α: Impact Parameter.
2:  Output: k clusters where each cluster contains several nodes of the set V.
3:  Pre-processing; /* Nodes set, Edges set, Attributes set*/
4:  Connection Extraction; /* Directed, Indirected and Disconnected connection- An
    Undirected, weighted graph(G1) and A Multi Attributed, weighted, and Undirected graph
    (G2) */
5:   Begin
6:     for every pair of vertices $V_n$ and $V_m$ in V where $n \neq m$
7:      Begin
8:        If $V_n \leftrightarrow V_m$ then
9:          Calculate Struct-similarity ($V_n$, $V_m$) in G1; using Eq. (5)
10:         Calculate Attribute-similarity ($V_n$, $V_m$) in G2; using Eq. (10)
11:     Else if $V_n \nleftrightarrow V_m$ then
12:         Calculate Struct-similarity ($V_n$, $V_m$) in G1; using Eq. (5)
13:       Calculate Attribute-similarity ($V_n$, $V_m$) in G2; using Eq. (10)
14:     Else /* $V_n \ldots V_m$ */
15:       Struct-similarity=0;
16:     Calculate Attribute-similarity ($V_n$, $V_m$) in G2; using Eq. (10)
17:     end for
18:   for every pair of vertices $V_n$ and $V_m$ in V where $n \neq m$
19:     Begin
20:     Calculate Hsim ($V_n$, $V_m$); using Eq. (11)
21:     Calculate Distance ($V_n$, $V_m$); using Eq. (12)
22:  End
23: Perform k-Medoids clustering /* Based on Distance Matrix, between any pairs of Nodes */
24: End.
```

3.3.1. Structural similarity

This section calculates the structural similarity between the two vertices of the graph according to the connection type between vertices, as shown in Algorithm 2. In similarity-based methods in heterogeneous networks, with only an absolute emphasis on the number of common neighbors, cannot calculate the structural similarity among pairs of nodes well. On the other hand, beyond direct relationships, also hidden relationships between any pair of vertices, such as indirect and disconnected connectivity, may contribute to the structural similarity calculation. First, the adjacency vector is calculated for each node of the G1 (line 4). Then, the union neighborhood set of the pairs of vertices (line 5) and the correlation between vectors is calculated to determine the correlation between two vertices (line 6). Finally, the structural similarity of the pair of vertices is obtained (line 7). The details of calculating the structural similarity of two nodes using the neighborhood of both nodes and their indirect interaction strength in three directly connected, indirectly connected, and disconnected states are described in the next section.

Algorithm 2: Pseudocode for Indirect and Direct Connected Structural Similarity

```
1:  Begin
2:  for each vertex $V_n$ in $V$ /* $v \in V$ */
3:    for each vertex $V_m$ in $V$
4:      Adjacency vector Calculate; using Eq. (1)
5:      Union adjacency Vector; using Eq. (2)
6:      Correlation Coefficient Calculate; using Eq. (3), (4)
7:      Structural similarity Calculate; using Eq. (5)
8:  End.
```

In the following, the method of calculating the structural similarity between directly and indirectly connected nodes is expressed. In most current techniques that consider the connection between nodes in the calculation of similarity, only paths with length two are considered in the indirectly connected type. Since paths with a length of more than two may contribute to the calculation of structural similarity in indirect connections, such paths are considered in the proposed method. The proposed adjacency vector in indirect nodes does not limit the path length. The adjacency vector of each node is calculated by Equation (1):

(1)

$$AV_n[m] =_{m\in v} \begin{cases} \sum_{i=1}^{d_n} w_{ni} & if\ n = m \\ \sum_{i=source}^{destination} w_{ni} & if\ vn, vm\ \ is\ indirected \\ w_{nm} & if\ vn, vm\ \ is\ directed \\ 0 & if\ vn, vm\ \ is\ disconnected \end{cases}$$

Where *n* is the node whose adjacency vector should be calculated. If the index number of the adjacency vector is equal to *n*, the sum of the weight of all edges entered into the node *n* is placed in this index. If the connection between *n* and *m* is indirect, the desired index value in the adjacency vector will be the sum of the weight of the shortest path between *n* and *m* in the simple graph. If *n* and *m* have a direct connection, then the weight of the direct edge is placed between them in the vector index. And if two nodes are disconnected, a zero value will be entered in the desired index. After calculating the adjacency vector of all nodes of the simple graph, the union neighborhood set between both indirectly and directly connected nodes is calculated based on Equation (2):

$$UNION_{nm} = \{z \mid (AV_n[z] > 0)\ or\ \ (AV_m[z] > 0)\} \tag{2}$$

To indicate the correlation between the pairs of nodes, the correlation coefficient between the union neighborhood set of the vectors is calculated by Equation (3):

$$corr_{nm} = \frac{\sum_{z\in UNION_{nm}} (AV_n[z] - \overline{AV_n})(AV_m[z] - \overline{AV_m})}{\sqrt{\sum_{z\in UNION_{nm}} (AV_n[z] - \overline{AV_n})^2}\sqrt{\sum_{z\in UNION_{nm}} (AV_m[z] - \overline{AV_m})^2}} \tag{3}$$

Where $\overline{AV_n}$, is the average value of the union neighborhood set of vector AV_n, which is obtained from Equation (4):

$$\overline{AV_n} = \frac{\sum_{z\in UNION_{nm}} AV_n[z]}{|UNION_{nm}|} \tag{4}$$

Finally, the structural similarity of any two nodes connected indirectly and directly is calculated using Equation (5):

$$sim(V_n, V_m)_{Struct} = (1 + W_{nm})(1 + corr_{nm}) \quad (5)$$

Where W_{nm} , is the weight of the common edge between two nodes *n* and *m* in the simple graph and $corr_{nm}$, is the correlation coefficient between them. Also, the structural similarity between disconnected nodes is assumed to be zero.

3.3.2. attribute similarity

There are different types of nodes in heterogeneous networks, each node in such a network can contain an *M* attribute, and each attribute can have a different set of *A* values. Since the goal is to calculate the hybrid similarity in such a network, the attribute of the nodes should be considered. For example, in a bibliographic network, one of the types of nodes is authors, and one of the attributes of nodes is the interest of each author in different research fields. As shown in Figure 3, an attribute is defined for each node, which contains four values (e.g., Data Mining, Data Base, Programming, and Machine Learning, which are four values for the interesting attribute).

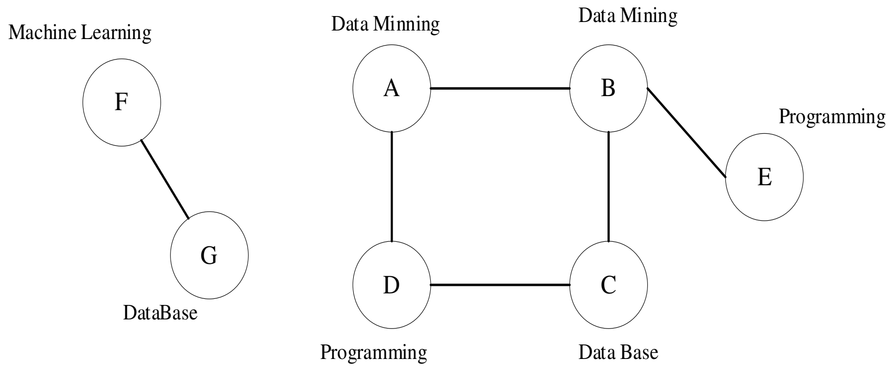


Fig.3. Graph with one attribute and four values

The attribute similarity, the like the structural similarity, is calculated based on three connection types. To calculate the attribute similarity and simplify the calculations, the *G2* is extracted from the sets of *V* and *A*. In a bipartite graph, there are two disjoint sets of nodes, such that the nodes of each set are not related, and only their connection is with the nodes of the opposite group, as shown in Figure 2-b. In all the calculations of this section and according to the connection types, the calculations of the attribute similarity will be done on the bipartite graph. The attribute similarity is responsible for calculating the attribute similarity between the two vertices of the graph, as shown in Algorithm 3. First, the adjacency vector is calculated for each node of the

G1 based on the *G2* (line 3). Then, the union neighborhood set of the pairs of vertices (line 4) and the correlation between vectors to determine the correlation between two vertices (line 5) is calculated. Finally, the attribute similarity of both nodes is obtained (line 6). In the next section, attribute similarity calculation is described in detail.

Algorithm 3: Pseudocode for Indirect, Direct and Disconnect Connected Attribute Similarity

```
1:  Begin
2:  Input: Bipartite Graph and Indirect, Direct and Disconnect Connected Nodes Set.
3:   Adjacency Vector Calculate; using Eq. (6)
4:   Union adjacency Vector; using Eq. (7)
5:  Correlation Coefficient Calculate; using Eq. (8), (9)
6:  Attribute similarity Calculate; using Eq. (10)
7:  End.
```

According to Algorithm 3, the adjacency vector of each node in *G2* is calculated by Equation (6):

$$AV_n[m] =_{m\in V} \begin{cases} \sum_{i=1}^{d_n} w_{ni} & if\ n = m \\ \sum_{i=1}^{M} common(Vn, Vm) * w_{attri} + w_{nm} & if\ Vn, Vm\ \ is\ directed \\ \sum_{i=1}^{M} common(Vn, Vm) * w_{attri} & if\ Vn, Vm\ \ is\ indirected \\ \sum_{i=1}^{M} common(Vn, Vm) * w_{attri} & if\ Vn, Vm\ \ is\ disconnected \end{cases} \tag{6}$$

Where *n* denotes the node whose adjacency vector should be calculated, if *m* is equal to *n*, the weight of all edges entered into the node *n* in the *G2* is placed in the m index. If *n* and *m* have a direct connection, then the weight of the common attribute edges between *n* and *m* in the *G2* is summed with the weight of the common edge between them in the simple graph. We consider the edge weight of each attribute is always considered as one. If the connection between *n* and *m* is indirect or disconnected, the value of the desired *m* index will be the sum of the weight of the common attribute edges in the *G2*.

After calculating the adjacency vector of all nodes, the union neighborhood set between both indirectly connected, directly connected, and disconnected is calculated based on Equation (7):

(7)

$$UNION_{nm} = \{z \mid (AV_n[z] > 0) \; or \; (AV_m[z] > 0)\}$$

A higher correlation between the union neighborhood set, $UNION_{nm}$, vectors AV_n and AV_m , demonstrates a higher structural similarity among nodes n and m. The correlation coefficient between the union neighborhood set of the vectors is calculated by Equation (8):

$$corr_{nm} = \frac{\sum_{z \in UNION_{nm}} (AV_n[z] - \overline{AV_n})(AV_m[z] - \overline{AV_m})}{\sqrt{\sum_{z \in UNION_{nm}} (AV_n[z] - \overline{A_n})^2} \sqrt{\sum_{z \in_{UNION_{nm}}} (AV_m[z] - \overline{AV_m})^2}} \tag{8}$$

Where $\overline{AV_n}$, is the average value of the union neighborhood set of vector AV_n, which is obtained from Equation (9):

$$\overline{AV_n} = \frac{\sum_{z \in UNION_{nm}} AV_n[z]}{|UNION_{nm}|} \tag{9}$$

In Equation (9), the fraction's numerator is the sum of the non-zero values of the *n^th^* node's adjacency vector. The fraction's denominator is the number of members union neighborhood set by the adjacency vectors of two nodes, n and m.

Finally, the attribute similarity between pairs of nodes based on the connection types will be calculated by Equation (10) as follows:

$$sim(V_n, V_m)_{attribute} = (1 + W_{nm})(1 + corr_{nm})/M \tag{10}$$

In Equation (10), W_{nm} is two nodes' common edge weight V_n and V_m in the simple graph, $corr_{nm}$, the correlation coefficient between them, and M is the number of attributes of the graph nodes.

3.3.3. Hybrid similarity and distance function

The overall similarity of both nodes with the combination of structural and attribute similarities are calculated by Equation (11):

$$Hsim(V_n, V_m) = \alpha . sim(V_n, V_m)_{Struct} + (1 - \alpha) . sim(V_n, V_m)_{attribute} \quad (11)$$

In Equation (11), the degree of influence of the two similarities is not the same. The α parameter is a weighting factor used to control the influence of both similarities, and, in advance, it must be in the range of [0,1] to be given. The suitable amount of α is the value that divides the graph into k clusters such that the nodes of each cluster have coherent communication structures and the same attribute values. In our method, based on the analysis of the results, the value of this coefficient is assumed to be 0.5, in which identical importance is given to structural and attribute similarities. After calculating the hybrid similarity measure according to connection types for performing the graph clustering process, the distance value for each pair of nodes in the graph is computed with Equation (12):

$$Distance(V_n, V_m) = \begin{cases} \dfrac{1}{Hsim(V_n, V_m)} & if\ Hsim > 0 \\ \infty & if\ Hsim = 0 \end{cases} \quad (12)$$

The distance value is the inverse of the similarity value. The smaller the distance between the nodes placed in a cluster, the better the clustering quality.

3.3.4. Graph clustering

K-Medoids algorithm is applied for graph clustering. K-Medoids is an iterative partitioning solution, as shown in Algorithm 4. We carry out the clustering process until the clusters converge. The number of clustering algorithm iteration and the cluster's number (k) is input to the proposed algorithm. The top k vertices with the maximum degree in V are selected as the k initial centres for the clusters (line 4). The rest of the nodes are assigned to each cluster according to their distance from the primary centroids (line 7). In each iteration of the algorithm, one node is selected from the remaining nodes to have the highest degree among the rest of the nodes (line 12). It is the new centroid of its cluster. The distance of the newly selected centre with all other graph nodes is calculated, and the clusters are updated. Next, the distance between each cluster's nodes and the centroid is computed. The total distances of all clusters are added together (lines 13-15). Suppose the obtained value is more suitable than the same value in the previous clustering. In that case, the new centroid is fixed, and the process continues (lines 16-18), else the centroid is removed, and the node with the next maximum degree is chosen, and the process will be repeated.

Algorithm 4: Pseudocode for K-Medoids Clustering

```
1:  Begin
2:  Input: $K$: number of clusters, $MaxIterationNumber$: The maximum number of iterations.
3:  Output: $K$ Clusters $c_1, c_2, \ldots, c_k$.
4:   $ClusterCentroid$ =Top k Maximum Degree vertices in $V$ set.
5:   remainingNodes= $V$ – ClusterCentroid.
6:  for (every $V_i$ in remainingNodes)
7:    Cluster[$i$]= min {Distance$_{(i,j)}$} $i, j$ for all centroids $j = 1 \ldots k$.
8:  while iterations <= $MaxIterationNumber$
9:        begin
10:       for each $v$ in remainingNodes
11:       begin
12:           Choice node with Maximum degree from the set of remainingNodes as newly
              Centroid.
13:           Calculate the distance of all the remainingNodes to the new centroid.
14:           Assign one node to a cluster that has a minimum distance from the centroid that
              Cluster.
15:           update total clusters.
16:           if (the sum of distances in all clusters is the minimum) then

17:                  update the ClusterCentroid.
18:           end
19:    iterations ++.
20: End.
```

4. Performance evaluation

This section validates the proposed solution using two real datasets, DBLP and Political blogs. Then, it describes the simulation parameters setup and performance metrics. Finally, a discussion of the simulation results follows it will provide.

4.1. Experimental setup

The experiments were performed on a 64-bit machine with a 2.80 GHz Intel Core i7 processor with 8 GB main memory and Windows 10 as an operating system. Python 3.9 is used as the open-source language to implement the suggested method. We compare the proposed method with the following three basic approaches to evaluate it. These methods have been fully simulated and implemented under the same conditions. We chose these methods because they calculate collaborative similarity using topological structures and features in undirected, multi-attribute, and weighted networks similar to ours. The following methods:

IGC-CSM [2]: A collaborative approach for clustering a weighted, multi-attribute, and undirected graph. This method computes topological similarity and attributes depending on the types of connection between nodes. The directly connected nodes' similarity is according to the similarity of Jaccard and the weight of the neighbors of the nodes. The structural and attribute similarities of nodes connected indirectly are the linear product of the structural similarity and the linear product of the attribute similarity of the two directly connected nodes in the path of the indirectly connected pair. This approach uses a shortest-path strategy to decrease the computation cost and search space. The K-medoids method is used to cluster the graph.

AR-Cluster [12]: A collaborative approach for graph clustering is based on the type of connection between nodes. Attracting and Recommending Degrees are used in this algorithm to compute the structural similarity. In addition, the K-medoids method is used to cluster the graph.

SAG-Cluster [24]: According to the type of connection of nodes, a cooperative approach is to cluster the graph with the K-Medoids framework. In calculating the structural similarity between directly connected nodes, the weight of all the edges with the neighboring nodes of each node is considered. Through the use of the classical Basel theorem and the maximum weighted average, they calculate the similarity between each indirectly connected pair.

In our experiments, we utilize two real datasets, DBLP and Political blogs.

Political blogs[1]: Political blogs included 1,490 blogs about United States politics, with 19,090 links among these web blogs. The attribute of each blog is its political leaning, the value of which is either liberal or conservative. In the experiments, the edge weight between blogs is considered one; also, one attribute with two liberal or conservative values for the nodes is considered.

DBLP[2]: We use a subset of DBLP bibliography information data. This network includes information on articles, citations to articles, information on authors, and author collaborations between them. The used sub-network was collected between 2004 and 2014. Our selected sub-network contains four research areas of Artificial Intelligence (AI), Information Retrieval (IR),

[1] http://www-personal.umich.edu/~mejn/netdata/

[2] https://www.aminer.org/aminernetwork

Data Mining (DM), and Data Base (DB). This network is a network weighted and multi attributes. In experiments, the attributes of the nodes are the authors' interest in different research fields. The number of co-authorships between authors is the edge weight between them. Each node has four attributes, and each attribute has a value. Looking at the datasets used according to the communication types among the nodes, the number of connections of various types is not the same. The number of indirectly connected links in Political blogs is more than the same type in the DBLP dataset. In DBLP, the number of disconnected links is more than the like in the Political blogs dataset.

4.2. Performance metric

We used the following performance measures to validate the proposed solution with other algorithms.

Density: Density is the ratio of the number of edges in a cluster to the number of edges in the entire graph. The proportions of all clusters are accumulated to assess their impact [2]. The density values lie in the range [0, 1]. The density is calculated by Equation (13):

$$Density(\{V_c\}_{c=1}^{k}) = \sum_{c=1}^{k} \frac{|\{(V_m,V_n)|V_m,V_n \in V_c\,,(V_m,V_n) \in E\}|}{|E|} \tag{13}$$

Where k is the number of clusters, and the c is each of the graph clusters, $|E|$ is the total number of edges in the graph and $|(V_m,V_n)|$, is the number of edges in cluster c.

Entropy: This metric is described to determine the relationships between vertices in terms of attributes [12]. A lower entropy means a better quality of clustering. The entropy value is in the range of [0, 1] and is expressed by Equation (14):

$$Entropy(\{V_c\}_{c=1}^{k}) = \sum_{c=1}^{M}\left(\frac{W_{attr_c}}{\sum_{s=1}^{M} W_{attr_s}} \sum_{i=1}^{k} \frac{|V_i|}{|V|}\, Entropy(attr_c, V_i)\right) \tag{14}$$

$$Entropy(attr_c, V_i) = -\sum_{n=1}^{n_c} Prcnt_{cin}\, log_2\, Prcnt_{cin}$$

Where i is each of clusters, $i= \{1, 2, ..., k\}$, W_{attr_c} , is the weight of the c^{th} attribute, n is attribute values, and n_c , is the number of attribute values. $Prcnt_{cin}$ is defined as the percentage of vertices in cluster j that have the value $attr_{cn}$ in the attribute $attr_c$.

Purity: This metric shows how many percentages of the nodes in a cluster have the same attributes. A higher value for this measure indicates better clustering performance. The purity value is in the range [0,1] and is calculated by Equation (15):

$$Purity(\{V_c\}_{c=1}^{k}) = \sum_{i=1}^{k} \frac{|V_i|}{|V|} Purity(V_i) \tag{15}$$

$$Purity(V_i) = Max_j(P_i(attr_j))$$

Where i is each of clusters, $i =\{1, 2,...,k\}$, the i^{th} cluster consists of V_i nodes, V is the whole number of graph nodes, and $P_i(attr_j)$ is the ratio of attribute j^{th} in the i^{th} cluster.

4.3. Experimental analysis

Simulation parameters are set in our proposed solution and other implemented algorithms, as shown in Table 3. α equals 0.5. Also, because specific instructions to achieve the maximum effective value of the *MaximumIteration* parameter are not defined, the appropriate value of this parameter, according to the analysis results, is assumed to be 45. The *MaximumIteration* parameter is the number of iterations of the clustering algorithm until the clusters converge.

Table 3. Setting simulation parameters

DataSets		**MaxIteration**	α
DBLP	**Political blogs**		
k=10	*k*=3	45	0.5
k=30	*k*=5	45	0.5
K=50	*k*=7	45	0.5
k=70	*k*=9	45	0.5

The quality of the results is evaluated using three criteria: density, entropy, and purity. The final results are presented as follows.

Figure 4 compares the density criterion for four approaches in the Political blogs dataset. In figure 4, the number of clusters is assumed to be 3, 5, 7, and 9, respectively. As the figure shows, when *k* increases in each approach, the cluster density decreases. In all cases, the *IGC-SCM* approach has a greater density than the other three. When *k* = 7 or 9, the proposed approach's density value is higher than in the *SAG-Cluster,* and *AR-Cluster* approaches. The density of the *AR-Cluster* approach is lower than other approaches in every case. The values of density of the *SAG-Cluster* and the proposed method are nearly close in *k*=5.

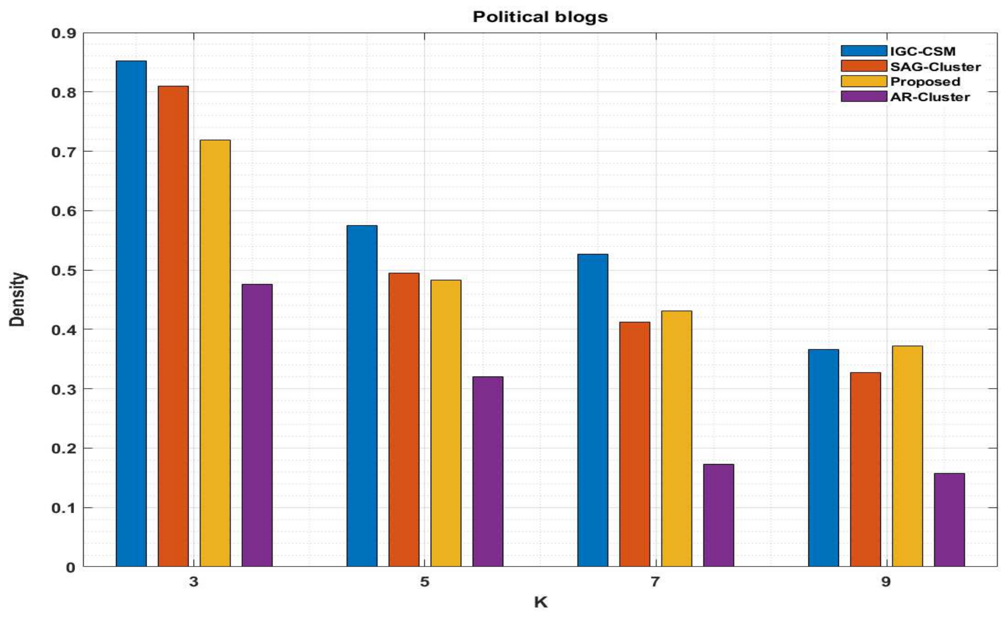


Fig4. Comparison of density value in Political blogs.

Entropy is used to determine the attribute relationships among nodes. A lower value of entropy means more homogeneity of the nodes of a cluster in terms of their attributes. Figure 5 compares the entropy criterion for four approaches on the Political blogs dataset with cluster numbers *k* = 3, 5, 7, and 9. The proposed method has the lowest entropy value in different values of *K*. We can infer that the proposed method is strictly considered attribute similarity. The entropy of the *SAG-Cluster* is better than *IGC-CSM* and *AR-Cluster*. *AR-Cluster* has a much higher entropy than the other three approaches in the above cluster numbers, which shows the weaker performance of this method.

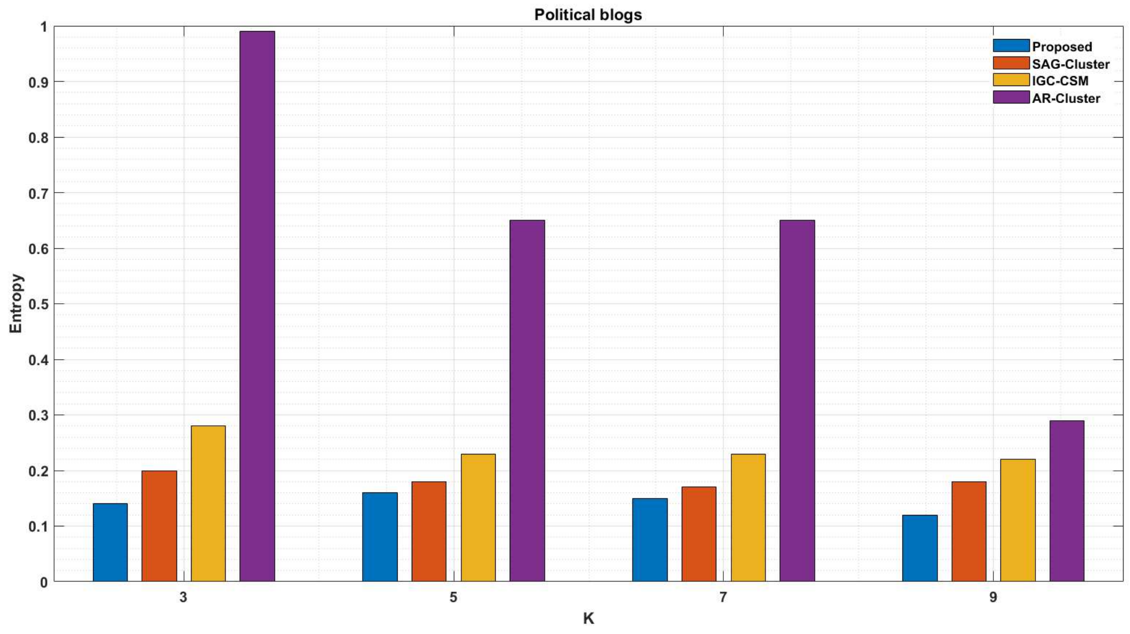


Fig 5. Comparison of entropy value in Political blogs.

Figure 6 compares the density criterion for four approaches on the DBLP dataset using cluster numbers 10, 30, 50, and 70. The density value of the *IGC-CSM* is the highest. While k = 10, the density value of the proposed method is lower than the density of the *SAG-Cluster* approach. The density values of the proposed method are higher than the *SAG-Cluster* and *AR-Cluster* when *k* = 30. The values of density of the *AR-Cluster* and the proposed method are nearly equal at *k*=70. The density values of the *SAG-Cluster* are higher than the proposed method when *k* =50,70.

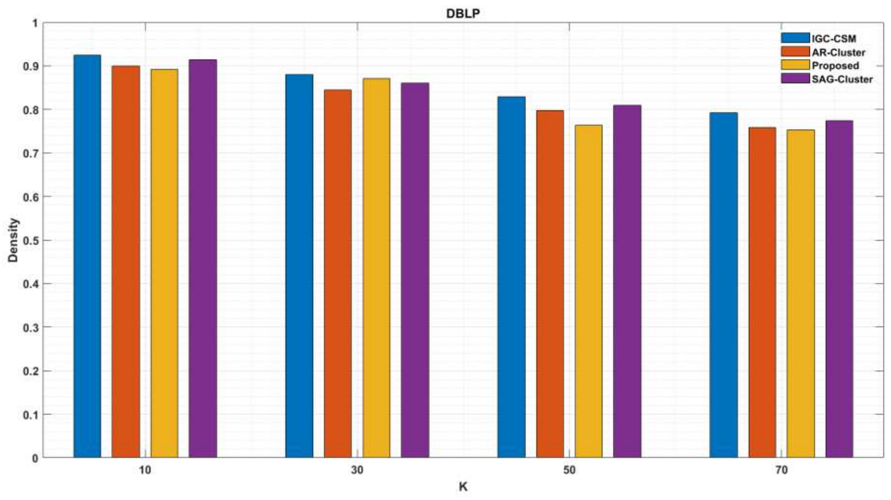


Fig6. Density value comparisons on DBLP.

Figure 7 compares the entropy criterion for four approaches on the DBLP dataset with cluster numbers $k = 10$, 30, 50, and 70. The entropy values of the *AR-Cluster*, *IGC-CSM*, and the *SAG-cluster* when $k = 10$, 30, or 50 are almost close. *SAG-Cluster* entropy is less than *AR-Cluster* and *IGC-CSM* when k reaches 70. The proposed method's entropy is less at different K values than the other three methods. We can infer that in the proposed method paying attention to the attributes of nodes is much more than the other methods.

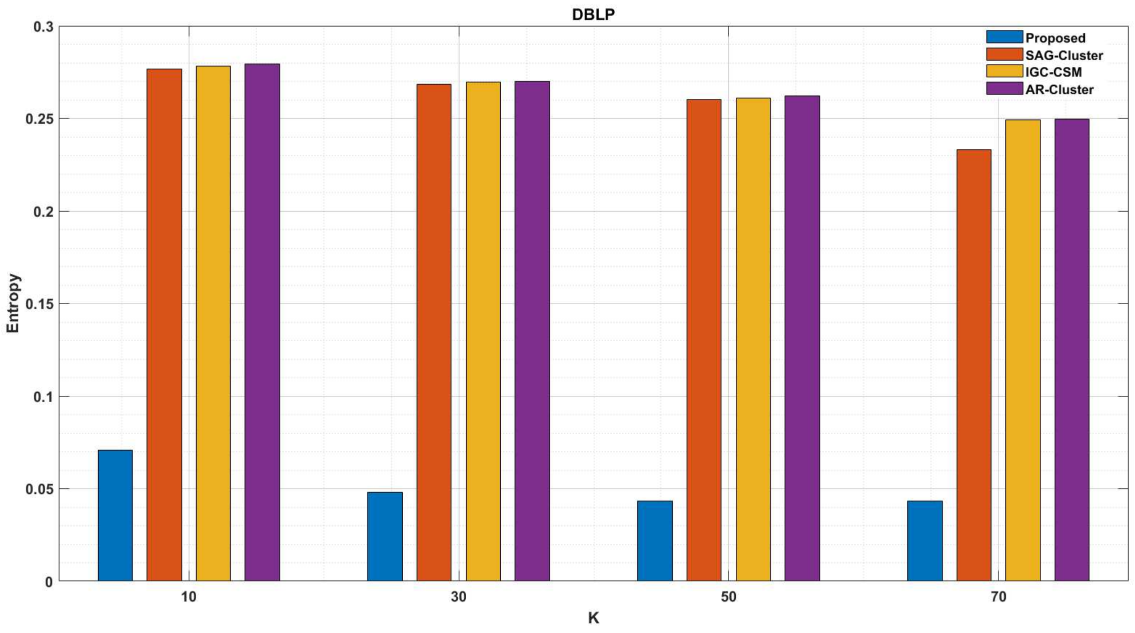


Fig7. Entropy value comparisons on DBLP.

Our proposed method's time complexity is quadratic, making it suitable for small and medium-sized graphs. Figure 8 shows the execution time of the proposed method in terms of the size of the graph based on the number of nodes in several examples on political blogs and DBLP data.

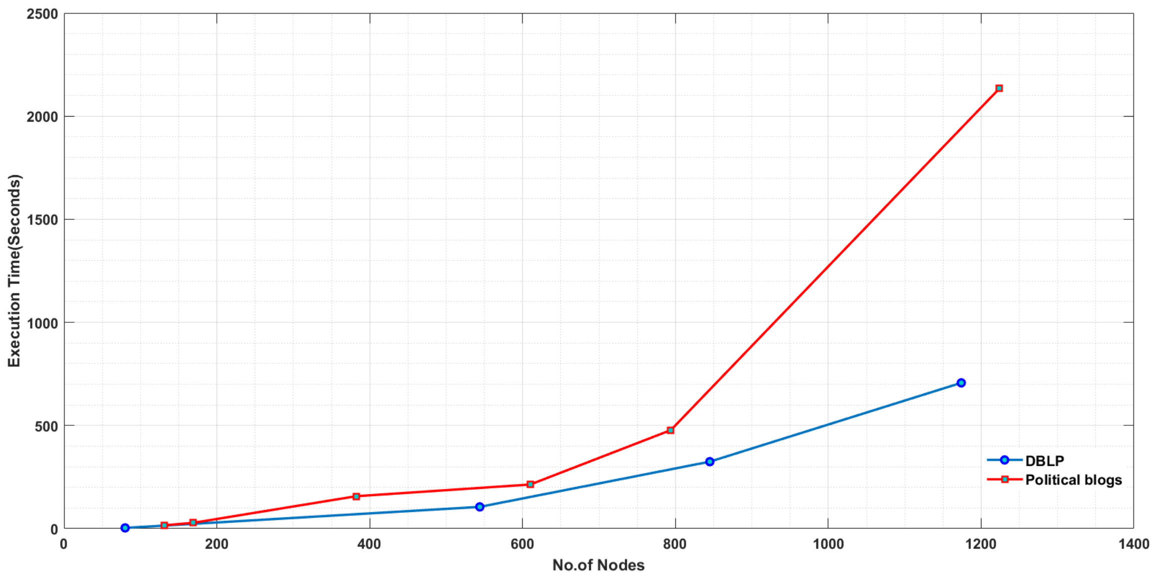


Fig8. An analysis of the proposed approach execution time on Political blogs and DBLP datasets

The execution time of the proposed approach is shorter than that of the other three approaches, especially in the Political blogs dataset, which has more indirect relationships. Since all three approaches calculate collaborative similarity based on the shortest path between indirectly connected nodes, this step increases the overall execution time in them. For example, the execution time of the proposed approach, according to Figure 8, on a subset of the Political blogs dataset with about 382 nodes is approximately 158 seconds, and the execution time of the *SAG-Cluster* approach on the same set is higher than 5400 seconds. Thus, the proposed approach has a superior runtime compared to other methods.

Figures 9 and 10 show a plot of density versus entropy. A line connects all points related to an algorithm. The direction of each line shows the treatment of the corresponding algorithm as the number of clusters increases. Arrowheads and tails indicate the minimum and maximum number of clusters [18]. The best performance is where the plot between density and entropy is in the upper left corner of the x-y plane, where density is the highest value and entropy is the lowest value. The quality of the proposed and the *SAG-Cluster* approaches on the DBLP dataset is quite effective compared to the other techniques, as shown in Figure 9. In the Political blogs dataset, the quality of the proposed approach is more effective than the three different approaches. The *AR-Cluster* approach is weaker than the comparative approaches, as shown in Figure 10.

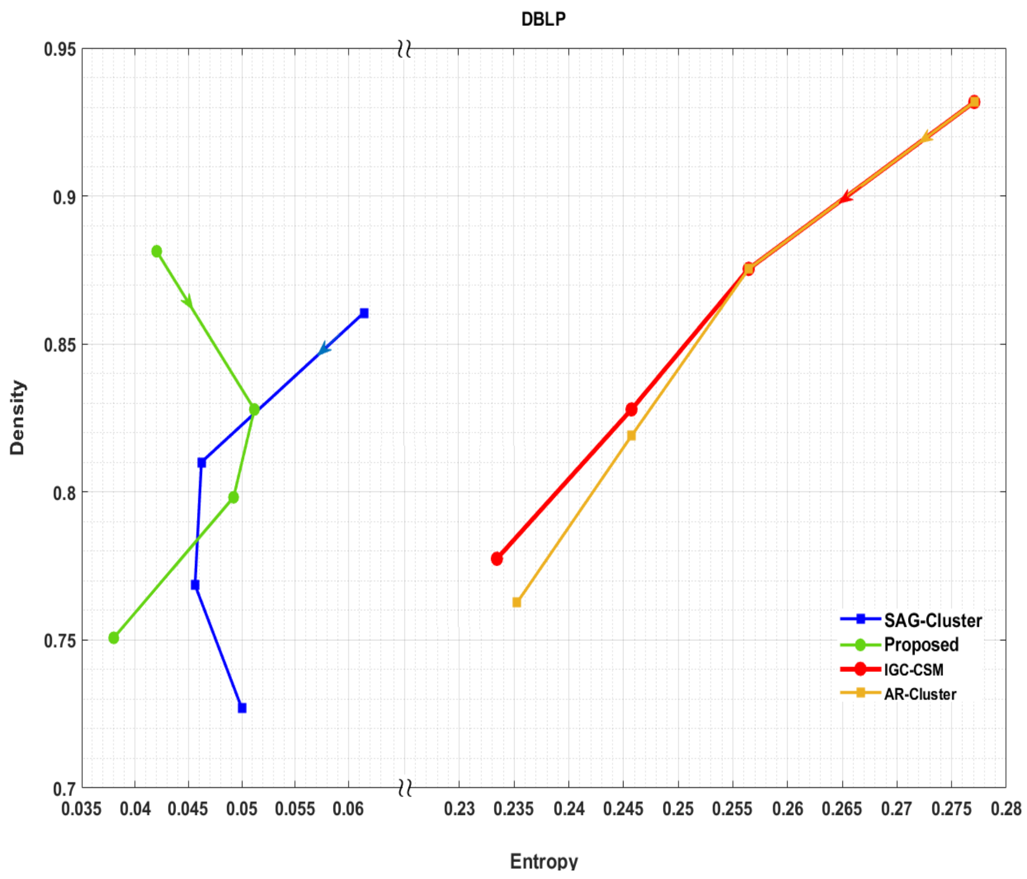


Fig9. The trade-off (density versus entropy). Analysis of the DBLP dataset

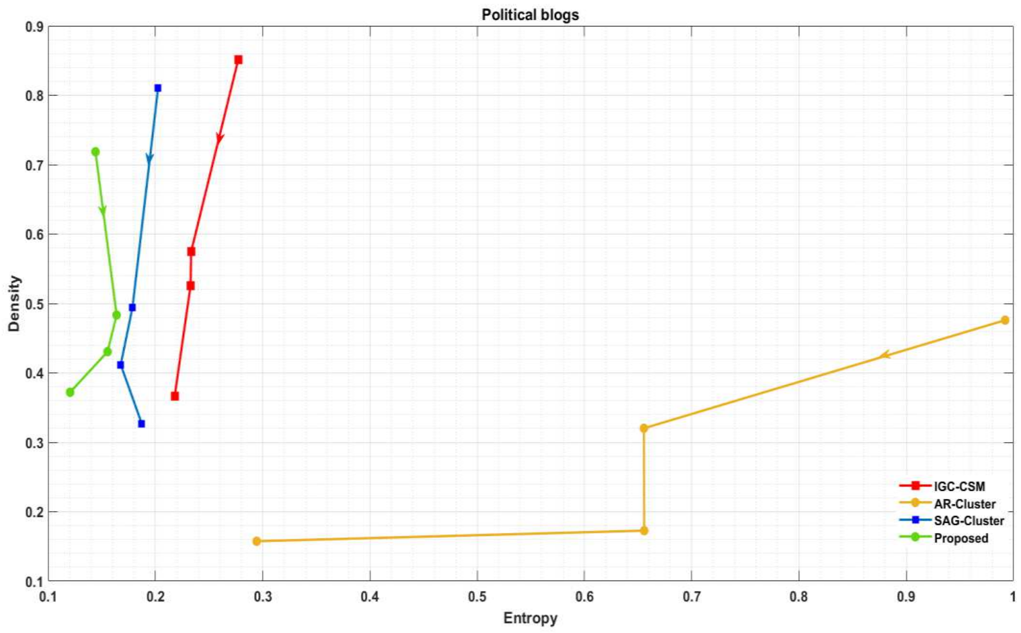


Fig10. The trade-off (density versus entropy). Analysis of Political blogs dataset

For further evaluation, we use the purity criterion. Our proposed algorithm is compared to three others in the purity criterion on Political blogs and DBLP, as shown in Figure 11. Experimental results show that the purity of the proposed method on both data sets is higher than other methods. Especially this value is much higher in the DBLP dataset. In Figure 11, in the Political blogs and DBLP datasets, respectively, purity is the average purity of clustering with k = 3, 5, 7, 9, and K = 10, 30, 50,70. A higher value for purity indicates a better clustering quality. In other words, the nodes of a cluster have more same attributes.

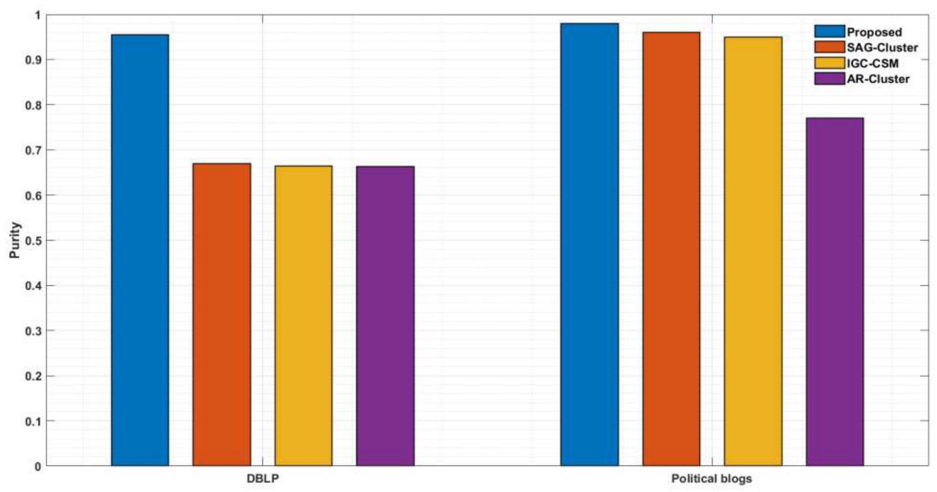


Fig11. Purity value comparisons on DBLP and Political blogs.

To evaluate the effectiveness of the proposed method, we considered it with the previous three methods according to density, entropy, execution time, and purity under two datasets of DBLP and Political blogs. Tables 4 and 5 show the performance of all four approaches under the two datasets used. According to these tables, in all experiments, the entropy of the proposed approach is lower than other approaches. When the number of nodes in the selected network is not high, the proposed approach gives a better density than entropy. By increasing the number of nodes, the entropy of the proposed approach will be better than the density. The purity metric in our method is always efficient and more than comparable methods.

Table 4. Comparison of different approaches on the Political blogs dataset

K	**Density**				**Entropy**				**Purity**			
	IGC-CSM	AR-Cluster	SAG-Cluster	Proposed	IGC-CSM	AR-Cluster	SAG-Cluster	Proposed	IGC-CSM	AR-Cluster	SAG-Cluster	Proposed
3	**0.851658**	0.476003	0.810209	0.718586	0.28	0.99	0.2	**0.14**	0.95026178	0.54973822	0.965968586	**0.976439791**
5	**0.574607**	0.320244	0.494764	0.483421	0.23	0.65	0.18	**0.16**	0.955497	0.811518	0.965969	**0.968586**
7	**0.526178**	0.172775	0.411867	0.430628	0.23	0.65	0.17	**0.15**	0.955497	0.777487	**0.965969**	**0.965969**
9	0.366492	0.157504	0.327225	**0.372164**	0.22	0.29	0.18	**0.12**	0.958115	0.929319	0.965969	**0.97644**

Table 5. Comparison of different approaches on the DBLP dataset

K	**Density**				**Entropy**				**Purity**			
	IGC-CSM	AR-Cluster	SAG-Cluster	Proposed	IGC-CSM	AR-Cluster	SAG-Cluster	Proposed	IGC-CSM	AR-Cluster	SAG-Cluster	Proposed
10	**0.924296**	0.899648	0.913732	0.892606	0.278128	0.279333	0.276846	**0.070947**	0.640237	0.63787	0.643787	**0.946746**
30	**0.880282**	0.84507	0.860915	0.861402	0.269545	0.269927	0.268256	**0.047961**	0.656805	0.656805	0.660355	**0.956213**
50	**0.829225**	0.797535	0.809859	0.764085	0.260844	0.262008	0.260155	**0.043375**	0.672189	0.671006	0.674556	**0.956213**
70	**0.792254**	0.758803	0.774648	0.75291	0.249405	0.249661	0.233026	**0.04333**	0.689941	0.688757	0.700592	**0.959763**

5. Conclusion

With the rapid development of social networks, data analysis of these networks to explore valuable information has become a significant research area. Clustering is one of the approaches to data analysis. The fundamental challenge in the clustering process is to consider the importance of both the structural relationships and the homogeneous characteristics of nodes. In this study, we proposed a hybrid clustering solution to predict links in heterogeneous information networks. It uses a combination of structural similarity and attribute similarity of nodes. Hence, we proposed a similarity measure according to the type of connection and correlation among the adjacency vectors of nodes. This measure in indirect nodes does not limit the path length. We evaluated the effectiveness of our solution under two real data sets. By comparing the proposed method with the existing methods, the simulation results showed that it is more effective in terms of entropy, purity, and execution speed. In addition, the cluster density is also preserved. We propose a quadratic time complexity method for small and medium-sized graphs. We will work on large-scale networks in the future, and we can also study the clustering of an information network with directed connections. Furthermore, we will develop a function to detect the convergence of the clustering algorithm based on density and entropy. In addition, we will follow the ability to find the best value for K based on the ratio of density to entropy without K being the input parameter of the clustering algorithm.

Declarations:

Ethical approval: All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki declaration and its later amendments or comparable ethical standards. This article does not contain any studies with human participants or animals performed by any of the authors.

Competing interests:

We certify that there is no actual or potential conflict of interest in relation to this article.

Authors' contributions:

Zahra Sadat sajjadi: Conceptualization, Data curation, Formal analysis, Methodology, Software, Validation, Writing - original draft. Mehdi esmaeili: Conceptualization, Data curation, Supervision. Mostafa Ghobaei-Arani: Conceptualization, Data curation, Supervision. Behrouz minaei: Conceptualization, Writing - review & editing

Funding:

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Availability of data and materials:

The data that support the findings of this study are available from the corresponding author, [author initials], upon reasonable request.

References

[1] Aggarwal, C. C. (ed.) (2011) Social Network Data Analytics. Boston, MA: Springer US.

[2] Nawaz, W. et al. (2015) "Intra graph clustering using collaborative similarity measure," Distributed and parallel databases, 33(4), pp. 583–603. https://doi: 10.1007/s10619-014-7170-x.

[3] Skabar, Andrew. 2017. "Clustering Mixed-Attribute Data Using Random Walk." Procedia Computer Science 108: 988–97. https://doi.org/10.1016/j.procs.2017.05.083.

[4] Roh, G.-P. and Hwang, S.-W. (2011) "Online clustering algorithms for semantic-rich network trajectories," Journal of computing science and engineering: JCSE, 5(4), pp. 346–353.

[5] https://doi: 10.5626/jcse.2011.5.4.346.

[6] Tian, Y., Hankins, R. A. and Patel, J. M. (2008) "Efficient aggregation for graph summarization," in Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD '08. New York, New York, USA: ACM Press.

[7] Fortunato, S. and Hric, D. (2016) "Community detection in networks: A user guide," arXiv [physics. Soc-ph]. Available at: http://arxiv.org/abs/1608.00163.

[8] Zhou, Yang, Hong Cheng, and Jeffrey Xu Yu. 2009. "Graph Clustering Based on Structural/Attribute Similarities." Proceedings of the VLDB Endowment International Conference on Very Large Data Bases 2 (1): 718–29. https://doi.org/10.14778/1687627.1687709.

[9] Cheng, H., Zhou, Y. and Yu, J. X. (2011) "Clustering Large Attributed Graphs: A Balance between Structural and Attribute Similarities," ACM Trans. Knowl. Discov. Data, 5.

[10] Sun, Y. et al. (2011) "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," Proceedings of the VLDB Endowment, 4, pp. 992–1003.

[11] Shi, C. et al. (2014) "HeteSim: A general framework for relevance measure in heterogeneous networks," IEEE transactions on knowledge and data engineering, 26(10), pp. 2479–2492. https://doi: 10.1109/tkde.2013.2297920.

[12] Li, X. et al. (2017) "Semi-supervised clustering in attributed heterogeneous information networks," in Proceedings of the 26th International Conference on World Wide Web. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee.

[13] Zhou, H. et al. (2017) "A graph clustering method for community detection in complex networks," Physica A, 469, pp. 551–562. https://doi: 10.1016/j.physa.2016.11.015.

[14] Yang, J., McAuley, J. and Leskovec, J. (2014) "Community detection in networks with Node Attributes," arXiv [cs.SI]. Available at: http://arxiv.org/abs/1401.7267.

[15] Lu, J., Gong, Z. and Lin, X. (2017a) "A novel and fast SimRank algorithm," IEEE transactions on knowledge and data engineering, 29(3), pp. 572–585. https://doi: 10.1109/tkde.2016.2626282.

[16] Shakibian, H. and Moghadam Charkari, N. (2017) "Mutual information model for link prediction in heterogeneous complex networks," Scientific reports, 7(1). https://doi: 10.1038/srep44981.

[17] Bai, L. et al. (2017b) "Fast graph clustering with a new description model for community detection," Information sciences, 388–389, pp. 37–47. https://doi: 10.1016/j.ins.2017.01.026.

[18] Huang, X., Cheng, H. and Yu, J. X. (2015) "Dense community detection in multi-valued attributed networks," Information sciences, 314, pp. 77–99. https://doi: 10.1016/j.ins.2015.03.075.

[19] Li, X. et al. (2022) "SCHAIN-IRAM: An efficient and effective semi-supervised clustering algorithm for attributed heterogeneous information networks," IEEE transactions on knowledge and data engineering, 34(4), pp. 1980–1992. https://doi: 10.1109/tkde.2020.2997938.

[20] Ghorbanzadeh, H. et al. (2021) "A hybrid method of link prediction in directed graphs," Expert systems with applications, 165(113896), p. 113896. https://doi: 10.1016/j.eswa.2020.113896.

[21] Zareie, A. and Sakellariou, R. (2020) "Similarity-based link prediction in social networks using latent relationships between the users," Scientific reports, 10(1), p. 20137. https://doi: 10.1038/s41598-020-76799-4.

[22] Wang, X. et al. (2021) "Link prediction in heterogeneous information networks: An improved deep graph convolution approach," Decision support systems, 141(113448), p. 113448. https://doi: 10.1016/j.dss.2020.113448.

[23] Jin, W., Jung, J. and Kang, U. (2019) "Supervised and extended restart in random walks for ranking and link prediction in networks," PloS one, 14(3), p. e0213857. https://doi: 10.1371/journal.pone.0213857.

[24] Berahmand, K. et al. (2022) "A new attributed graph clustering by using label propagation in complex networks," Journal of King Saud University - Computer and Information Sciences, 34(5), pp. 1869–1883. https://doi: 10.1016/j.jksuci.2020.08.013.

[25] Agrawal, S. and Patel, A. (2021) "SAG Cluster: An unsupervised graph clustering based on collaborative similarity for community detection in complex networks," Physica A, 563(125459), p. 125459. https://doi: 10.1016/j.physa.2020.125459.

[26] Kumar, A. et al. (2020) "Link prediction in complex networks based on Significance of Higher-Order Path Index (SHOPI)," Physica A, 545(123790), p. 123790. https://doi: 10.1016/j.physa.2019.123790.

[27] Kumar, A. et al. (2019) "Level-2 node clustering coefficient-based link prediction," Applied Intelligence, 49(7), pp. 2762–2779. https://doi: 10.1007/s10489-019-01413-8.

[28] Ghasemi, S. and Zarei, A. (2022a) "Improving link prediction in social networks using local and global features: a clustering-based approach," Progress in artificial intelligence, 11(1), pp. 79–92. https://doi: 10.1007/s13748-021-00261-3.

[29] Lande, D. et al. (2020) "Link prediction of scientific collaboration networks based on information retrieval," World wide web, 23(4), pp. 2239–2257. https://doi: 10.1007/s11280-019-00768-9.