Gradient Descent Clustering with Regularization to Recover Communities in Transformed Attributed Networks

Soroosh Shalileh[1,2][0000−0001−6226−4990]

[1] Laboratory of Artificial Intelligence for Cognitive Sciences (AICS), HSE University;
[2] Center for Language and Brain (CLB), HSE University, Moscow, RF.
sr.shalileh@gmail.com

Abstract. Community detection in attributed networks aims to recover clusters in which the within-community nodes are as interconnected and as homogeneous as possible, while the between-communities nodes are as disconnected and as heterogeneous as possible. The current research proposes a straightforward data-driven model with an integrated regularization term to recover communities. For further improvement of the quality of detected communities we also propose a softmax-scaled-dot-product to transform the data spaces into more cluster-friendly data spaces. We adopt the gradient descent optimization strategy to optimize our proposed clustering objective function. We compare the performance of the proposed method using both real-world and synthetic data sets with three state-of-art algorithms. Our results showed that the proposed method obtains promising result.

Keywords: Attributed Network · Feature-Rich Network · Community Detection · Gradient Descent · Clustering.

1 Introduction

Community Detection (CD) is a popular topic in context of social and complex networks analysis. The current research focuses on detecting communities in attributed networks. An attributed network is data structure in which, in addition to the conventional between-nodes links, each node is associated with a set of attributes. The objective of CD methods, in such a data structure, is to recover communities (clusters) using both sources of data, namely, the links and attributes information, such that the within-community nodes are as interconnected and as homogeneous as possible, while the between-communities nodes are as disconnected and as heterogeneous as possible.

Various CD methods have been proposed in the literature. For instance, in the period from 2020-2023, three comprehensive review papers were published on this subject [1–3]. The authors of [3], the most recent review, focused on the healthcare applications of CD. Prior to that, in the middle of 2021, Al-Andoli et al. [2] provided a more comprehensive review of CD methods, focusing on deep

learning and meta-heuristic approaches. Before these two reviews, in the middle of 2020, another comprehensive review was published [1]. In our opinion, this review introduced a more comprehensive taxonomy of the previous methods, which is still valid and applicable to even very recent developments, such as [4] and [5]. Thus we adopted its taxonomy.

In this taxonomy, concerning how the two sources of data are fused, the methods are categorized into: (i) early fusion, where the link and attribute data are fused before the cluster recovery process starts, (ii) simultaneous fusion, where the two sources of data are fused during the cluster recovery process, and (iii) late fusion, where the cluster recovery results of the two sources are fused after the process. We think the simultaneous fusion methods can be further classified into (ii-a) theory-based and (ii-b) data-driven methods. While (ii-a) derives a probability distribution of the process that generated the data, (ii-b) deals with the data as is. This research belongs to (ii-b), i.e, the data-driven simultaneous fusion category.

To be more precise, the current research continues the research line started in [6]. In that work, inspired by the triumph and recent advancements of deep learning (DL) methods, the authors adopted gradient descent (GD) optimization, the working horse of DL, for the task of clustering tabular data. Their results showed that, in the majority of the experiments, their proposed method, i.e., gradient descent clustering (GDC), had an edge over other benchmark clustering methods. Later, in [7], they utilized a non-summable data-driven model [8], and extended GDC to the attributed networks. However, that extension, in its simplest version did not lead to promising results, and thus they proposed a filter mechanism to cull out the misleading objects during GDC and to improve the quality of the results.

This research, unlike the methods reviewed in [2], pursues another direction to improve the performance of the GDC method extended to attributed networks. More precisely, we proposed two modifications to GDC: (a) we integrate a regularization term into our clustering objective function, to impose Occam's razor; and (b) we used a softmax-scaled-dot-product to transform the data spaces into more cluster-friendly spaces. We empirically evaluated and compared the performance of our proposed methods with three state-of-the-art (SOTA) algorithms using five real-world and 320 synthetic data sets.

2 Methodology

2.1 Softmax-scaled-dot-product to transform the data space

Given (an) $n \times m$ data matrix $Y = (y_{ij})$, let us denote its i-th row with $\mathbf{y}_i$, we transform Y to an $n \times n$ matrix $Y^{'} = (y^{'}_{ij})$ such that:

$$y^{'}_{ij} = \psi(\frac{< \mathbf{y}_i, \mathbf{y}_j >}{|m|}) \tag{1}$$

where $\psi(y_{ij}) = \frac{e^{(y_{ij})}}{\sum_{j=1}^{m} e^{(y_{ij})}}$ is the softmax function and the division by $|m|$ scales down the dot product to avoid numerical instabilities (and gradient explosions).

The rows of the transformed matrix, $Y^{'}$, now express the probability of relation of the rows with one another. We expect that this transformed space manifests the relationship between the data points more clearly, and the data points will be more clustering-friendly.

2.2 Notation and problem formulation

Let I represents the set of N nodes, we define an attributed network (AN) over this set as $D = \{X, A\}$, where $X \in \mathbb{R}^{N \times V}$ represents the features matrix, V is the number of features, and $A \in \mathbb{R}^{N \times N}$ represents the adjacency matrix. We aim to partition D into K crisp clusters using both sources of information, such that the within-group nodes are as interconnected and homogeneous as possible. To this end, each cluster, $\mathbf{s}_k$, is associated with its center in the feature space $\mathbf{c}_k$, and in the network space $\boldsymbol{\lambda}_k$, to form the set of centers in the feature space, $C = \{\mathbf{c}_k\}_{k=1}^{K}$, and in the network space, $\Lambda = \{\boldsymbol{\lambda}_k\}_{k=1}^{K}$, and the set of clusters, $S = \{\mathbf{s}_k\}_{k=1}^{K}$. Thus, we formulate the clustering objective function as follows:

$$J(D, C, \Lambda) = \sum_{i=1}^{N} \sum_{k=1}^{K} \rho f(\mathbf{c}_k, \mathbf{x}_i) + \xi h(\boldsymbol{\lambda}_k, \mathbf{a}_i) + \tau ||\mathbf{c}_k||_2^2 + \tau ||\boldsymbol{\lambda}_k||_2^2 \qquad (2)$$

where $f(.) : \mathbb{R}^V \longrightarrow \mathbb{R}$ ($h(.) : \mathbb{R}^N \longrightarrow \mathbb{R}$) represents a generic (continuous) distance function that will be used to measure the distance between the i-th data point in the feature (network) space and the corresponding centroid. The coefficients $\rho, \xi \in [0, 1]$ are meant to adjust the trade between the two data sources during the clustering procedure. In this research, we merely fix them to unity. The terms $||\mathbf{c}_k||_2^2$ and $||\boldsymbol{\lambda}_k||_2^2$ are $L2$ norm regularization terms; they are added to impose Occam's razor on the centroids' values, that is, to encourage smaller values of the centers (and decrease their variance). The coefficient $\tau \in [0, 1]$ controls the strength of the regularization terms. The larger its value, the higher the impact of the regularization term. We postponed applying other regularization terms such as $L1$ norm, Elastic Net and separating this term per each data source to our future study.

By applying the softmax-scaled-dot-product transformation, Eqn. (1), a transformed attributed network (TAN) will be obtained, i.e., $D^{'} = \{X^{'}, A^{'}\}$, where $X^{'} \in \mathbb{R}^{N \times N}$ represents the transformed features matrix, and $A^{'} \in \mathbb{R}^{N \times N}$ represents the transformed adjacency matrix. A similar clustering objective function (as per Eqn. (2)) can be defined. In the remainder of this section for the sake of notational brevity we avoid define this objective function and will be using Eqn. (2), unless stated.

2.3 Optimization of the clustering objective function

To optimize the clustering objective function, Eqn. (2), or its counterpart in TAN, we assume that the distance functions, $f(.), h(.)$, are continuous and dif-

ferentiable, and therefore we concentrate on the first-order derivative-based optimization algorithm, namely, gradient descent (GD) algorithm. GD is an iterative approach, therefore, we need to update our notation to demonstrate the concept of iterations. We did that by adding subscript (t). More precisely, we denote the set of centers in the feature space, in the network space, and the set of detected clusters, at iteration (t), with $C^{(t)} = \{\mathbf{c}_k^{(t)}\}_{k=1}^K$, $\Lambda^{(t)} = \{\boldsymbol{\lambda}_k^{(t)}\}_{k=1}^K$ and $S^{(t)} = \{\mathbf{s}_k^{(t)}\}_{k=1}^K$, respectively.

Our adoption of GD to detect clusters in AN or TAN, consists of three components: (i) a criterion to assign the objects to the communities (community assignment), (ii) a rule to update cluster centers, and (iii) a stop condition.

The first component (i), the community assignment criterion, is explained in Eqn. (3):

$$\underset{k}{argmin} f(\mathbf{x}_i, \mathbf{c}_k^{(t)}) + h(\mathbf{a}_i, \boldsymbol{\lambda}_k^{(t)}) < f(\mathbf{x}_i, \mathbf{c}_j^{(t)}) + h(\mathbf{a}_i, \boldsymbol{\lambda}_j^{(t)}), \forall j \neq k. \tag{3}$$

According to this criterion, at each iteration, the i-th vertex will be assigned to the k-th cluster for which the total sum of the distances between the i-th data point in the feature space, $\mathbf{x}_i$ and the corresponding center, $\mathbf{c}_k$, and the i-th data point vector, $\mathbf{a}_i$ and the corresponding center, $\boldsymbol{\lambda}_k$, is minimum.

The rules to update the centers (ii), in the feature and network spaces, are explained in equations (4) and (5), respectively:

$$\mathbf{c}_k^{(t+1)} = \mathbf{c}_k^{(t)} - \alpha[\nabla_{\mathbf{c}_k^{(t)}} f(\mathbf{x}_i, \mathbf{c}_k^{(t)}) + 2\tau\mathbf{c}_k] \tag{4}$$

and

$$\boldsymbol{\lambda}_k^{(t+1)} = \boldsymbol{\lambda}_k^{(t)} - \alpha[\nabla_{\boldsymbol{\lambda}_k^{(t)}} h(\mathbf{x}_i, \mathbf{c}_k^{(t)}) + 2\tau\boldsymbol{\lambda}_k] \tag{5}$$

where α denotes the step size, and $\nabla_{\mathbf{c}_k^{(t)}}$ is the gradient vector of the distance metric, $f(.)$, w.r.t the k-th feature center at iteration (t) which is evaluated with the data point $\mathbf{x}_i$. And $2\tau\mathbf{c}_k$ represents the gradient of the $L2$ norm regularization term. Similar description is valid for Eqn. (5). By equating τ to zero, the vanilla gradient descent update rule will be obtained. Using other values of τ in the range $(0, 1]$ adjusts the impact of the regularization terms, i.e., the larger the values of τ, the larger its impact. We treated τ as a hyper-parameter and empirically studied its impact in Sec. 4 to propose a default value for it.

Our preliminary experimental results were aligned with the results of [9], and showed that the on-line update rule is more efficient than the batch version of the GD algorithm; therefore, in the rest of this paper, we update the centroids in an on-line fashion.

In principles, there are various options for the last component of our proposed method (iii), i.e., the stop condition, for instance, applying first- and second-order necessary optimality conditions, etc., in this work, we relied on reaching the predefined maximum number of iterations. We treated it as a hyperparameter and empirically scrutinized its impact in Sec. 4 to propose a default value.

In the current research, to prove the efficiency of the introduced regularization and the data transformation, we proceed with the two versions of our proposed method. In the first version, we simply apply GD to optimize Eqn. (2) with $\tau = 0$. We called this version Gradient Descent Clustering in Attributed Network (GDCinAN). Clearly, GDCinAN represents the vanilla adoption of gradient descent clustering without any regularization or data space transformation, and forms a baseline for our empirical studies. In the second version we adopt GD to TAN with regularization. We called this version Gradient Descent Clustering with Regularization in Transformed Attributed Network (GDCRinTAN). We outline our proposed clustering methods in Algorithm (1).

Algorithm 1: GDCinAN and GDCRinTAN

Input: $D = \{X, A\}$: AN or $D^{'} = \{X^{'}, A^{'}\}$: TAN and K: number of clusters.
Hyperparameters: α: step size; T: maximum number of iterations; τ: regularization strength coefficients.
Result: $S = \{\mathbf{s}_k^{(t)}\}_{k=1}^{K}$ % set of K binary cluster membership vectors;
$C = \{\mathbf{c}_k^{(t)}\}_{k=1}^{K}$ % set of K centroids in feature space;
$\Lambda = \{\boldsymbol{\lambda}_k^{(t)}\}_{k=1}^{K}$ % set of K centroids in network space.
Initialize: Randomly initialize C, Λ and S.
for $t \in Range(T)$ **do**
  for $(\mathbf{x}_i, \mathbf{a}_i) \in D$ **do**
    find k using Eqn. (3) and set i-th entry of the $\mathbf{s}_k^{(t)}$ to one;
    update the centroids using the equations (4) and (5);
  end
end

We implemented our proposed methods using JAX, an automatic differentiation library [10]. The source code of the proposed methods are publicly available at https://github.com/Sorooshi/GDCRinTAN/tree/main.

3 Experimental setting

3.1 Competitors

We compared the performance of our proposed methods with DMoN [11], EVA [12], and KEFRIN [13]. DMoN is a clustering method based on graph convolutional networks and modularity criterion. EVA also utilizes modularity to model network data and the purity equation to model categorical features. KEFRIN is a meaningful extension of K-Means clustering to feature-rich networks.

We use Adjusted Rand Index [14], ARI as the evaluation metric. The closer the ARI to one, the better is the match between the recovered clusters and the ground truth.

3.2 Data sets

We compared the performance of the methods using synthetic and real-world data sets. We provide the summary of the five real-world data sets in Table 1.

Table 1: Real world data sets: the symbols N, E, and F represent the number of nodes, the number of links, and the number of attributes, respectively.

Name	Nodes	Edges	Features	Number of Communities	Ground Truth	Ref.
Malaria HVR6	307	6526	6	2	Cys Labels	[15]
Lawyers	71	339	18	6	Derived out of office and status features	[16, 17]
Parliament	451	11646	108	7	Political parties	[18]
COSN	46	552	16	2	Region	[19]
SinaNet	3490	30282	10	10	Users of same forum	[20]

To generate the synthetic attributed networks we applied the mechanism introduced in [8], which was later was applied in several works such as [13,21]. It ought to be added that we generated two types of features, namely, categorical, and mixed-scale features. Refer to any of the earlier-mentioned papers for more details on the mechanism applied to generate synthetic data. Each attributed network with categorical or mixed scale features had eight settings and each setting was repeated ten times, leading to 320 such synthetic data sets.

4 Study the impact of the hyperparameters of the proposed methods

4.1 Step size

We studied the impact of step size on the performance of our proposed methods and reported the results in Table 2.

Considering the number of wins, one can conclude that step size equal to 0.01 is the most suitable value for GDCinAN. Regarding GDCRinTAN, although this value also won four settings, however, a closer look at the results reveals that a step size equal to 0.1 is a better default value for this version of our proposed clustering method.

4.2 Regularization strength coefficient

We reported the impact various regularization strength coefficients, τ, on the performance of GDCRinTAN in Table 3. These results imply that the best performance GDCRinTAN is obtained when $\tau = 0.001$.

4.3 Impact of the number of iterations

We studied the impact of the number of iterations on the performance of our proposed methods and reported the results in Table 4. Considering the number

Table 2: Impact of step size (α) with fixed $n\_init = max\_iter = 10$. The best results are bold-faced.

Method	Configuration p, q, ζ	$\alpha = 0.0001$ ARI	$\alpha = 0.001$ ARI	$\alpha = 0.01$ ARI	$\alpha = 0.1$ ARI
GDCinAN	(0.6, 0.2, 0.6)	0.854 ± 0.104	0.893 ± 0.015	**0.899 ± 0.002**	0.834 ± 0.131
	(0.6, 0.2, 0.8)	0.895 ± 0.068	0.913 ± 0.014	**0.925 ± 0.003**	0.781 ± 0.145
	(0.6, 0.4, 0.6)	**0.175 ± 0.035**	0.003 ± 0.001	0.003 ± 0.001	0.003 ± 0.001
	(0.6, 0.4, 0.8)	**0.174 ± 0.091**	0.005 ± 0.001	0.004 ± 0.002	0.004 ± 0.001
	(0.8, 0.2, 0.6)	0.959 ± 0.032	0.966 ± 0.028	**0.968 ± 0.026**	0.951 ± 0.016
	(0.8, 0.2, 0.8)	0.942 ± 0.065	**0.967 ± 0.016**	0.949 ± 0.057	0.954 ± 0.059
	(0.8, 0.4, 0.6)	**0.774 ± 0.085**	0.762 ± 0.095	0.727 ± 0.153	0.551 ± 0.221
	(0.8, 0.4, 0.8)	0.817 ± 0.061	**0.838 ± 0.007**	0.767 ± 0.163	0.645 ± 0.204
GDCinTAN	(0.6, 0.2, 0.6)	0.969 ± 0.056	0.916 ± 0.033	**1.000 ± 0.000**	**1.000 ± 0.000**
	(0.6, 0.2, 0.8)	0.893 ± 0.065	0.894 ± 0.035	**0.999 ± 0.003**	**0.999 ± 0.002**
	(0.6, 0.4, 0.6)	0.768 ± 0.112	0.595 ± 0.082	0.878 ± 0.086	**0.903 ± 0.075**
	(0.6, 0.4, 0.8)	**0.707 ± 0.184**	0.318 ± 0.028	0.518 ± 0.019	0.669 ± 0.213
	(0.8, 0.2, 0.6)	0.953 ± 0.045	0.960 ± 0.064	**1.000 ± 0.000**	0.940 ± 0.092
	(0.8, 0.2, 0.8)	0.915 ± 0.064	**0.998 ± 0.003**	0.971 ± 0.059	0.961 ± 0.061
	(0.8, 0.4, 0.6)	0.860 ± 0.076	**0.927 ± 0.039**	0.905 ± 0.150	0.889 ± 0.169
	(0.8, 0.4, 0.8)	0.933 ± 0.087	0.935 ± 0.063	**1.000 ± 0.000**	**1.000 ± 0.000**

Table 3: GDCRinTAN: impact of τ using ARI values with $n\_init = T = 10$ and the step size $\alpha = 0.1$. The best results are bold-faced.

Configuration p, q, ζ	$\tau = 0.0001$	$\tau = 0.001$	$\tau = 0.01$	$\tau = 0.1$	$\tau = 0.3$	$\tau = 0.9$
(0.6, 0.2, 0.6)	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	0.978 ± 0.047	0.739 ± 0.242
(0.6, 0.2, 0.8)	**0.999 ± 0.002**	**0.999 ± 0.002**	**0.999 ± 0.002**	0.998 ± 0.005	0.955 ± 0.059	0.593 ± 0.371
(0.6, 0.4, 0.6)	0.909 ± 0.067	**0.918 ± 0.066**	0.911 ± 0.072	0.792 ± 0.154	0.581 ± 0.283	0.427 ± 0.334
(0.6, 0.4, 0.8)	0.966 ± 0.026	**0.974 ± 0.024**	0.669 ± 0.214	0.819 ± 0.149	0.715 ± 0.292	0.485 ± 0.294
(0.8, 0.2, 0.6)	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	**1.000 ± 0.000**	0.991 ± 0.021	0.975 ± 0.033
(0.8, 0.2, 0.8)	0.992 ± 0.023	**1.000 ± 0.000**	0.957 ± 0.066	0.865 ± 0.106	0.848 ± 0.074	0.823 ± 0.068
(0.8, 0.4, 0.6)	**0.995 ± 0.012**	0.977 ± 0.021	0.979 ± 0.039	0.739 ± 0.104	0.792 ± 0.115	0.741 ± 0.225
(0.8, 0.4, 0.8)	0.987 ± 0.040	0.987 ± 0.040	**1.000 ± 0.000**	0.999 ± 0.004	0.986 ± 0.026	0.767 ± 0.159

of wins, one can conclude that one iteration is sufficient for GDCinAN. The best performance of GDCRinTAN was obtained when we used 100 iterations. However, a closer look at the results implies that even ten iterations usually led to acceptable performance; and selecting the final best value should be decided by the user according to their preferences considering the speed and accuracy trade-off.

4.4 Tuned hyperparameters

We propose using a step size of 0.001 for GDCinAN and 0.1 for GDCRinTAN. To obtain the best performance for GDCRinTAN, the regularization strength coefficient, τ, is 0.001. For the number of iterations, the situation is less straightforward. More precisely, our experiments showed that any number between one to ten can be adopted for GDCinAN. And any number between ten and 100

Table 4: Studying the impact of the number of iterations using ARI values with tuned hyperparameters. The best results are bold-faced.

Configuration	GDCinAN			GDCRinTAN		
p, q, ζ	$T = 1$	$T = 10$	$T = 100$	$T = 1$	$T = 10$	$T = 100$
(0.6, 0.2, 0.6)	0.840 ± 0.098	0.854 ± 0.104	**0.856 ± 0.030**	0.982 ± 0.037	**1.000 ± 0.000**	**1.000 ± 0.000**
(0.6, 0.2, 0.8)	0.838 ± 0.045	0.895 ± 0.068	**0.903 ± 0.004**	0.966 ± 0.041	0.992 ± 0.002	**0.999 ± 0.002**
(0.6, 0.4, 0.6)	**0.227 ± 0.042**	0.175 ± 0.035	0.005 ± 0.002	0.852 ± 0.070	**0.918 ± 0.002**	0.909 ± 0.062
(0.6, 0.4, 0.8)	**0.304 ± 0.054**	0.174 ± 0.091	0.017 ± 0.004	0.943 ± 0.022	0.918 ± 0.066	**0.966 ± 0.028**
(0.8, 0.2, 0.6)	0.932 ± 0.060	**0.959 ± 0.032**	0.937 ± 0.041	0.999 ± 0.001	0.974 ± 0.024	**1.000 ± 0.000**
(0.8, 0.2, 0.8)	**0.953 ± 0.050**	0.942 ± 0.065	0.940 ± 0.030	0.986 ± 0.024	**1.000 ± 0.000**	0.992 ± 0.023
(0.8, 0.4, 0.6)	**0.883 ± 0.064**	0.774 ± 0.085	0.757 ± 0.003	0.980 ± 0.021	0.977 ± 0.021	**0.986 ± 0.020**
(0.8, 0.4, 0.8)	**0.833 ± 0.080**	0.817 ± 0.061	0.789 ± 0.106	0.995 ± 0.014	0.987 ± 0.040	**1.000 ± 0.000**

can be adopted for GDCRinTAN. We took a closer look to the performance of GDCRinTAN during the optimization procedure, and we realized that setting the number iterations to 20 usually leads to acceptable results. Therefore, in the rest of our experiments we set this number to 20.

5 Experimental results

5.1 Comparison over real-word data sets

We compared the performance of the proposed methods using five real-world data sets with three SOTA algorithms and reported the results in Table 5.

Table 5: Real-world data sets comparison: average values of ARI are presented over 10 random initialization. The best results are boldfaced.

data set	DMoN	EVA	KEFRiN	GDCinAN	GDCRinTAN
HRV6	0.64 ± 0.00	0.04 ± 0.00	**0.69 ± 0.38**	0.51 ± 0.18	0.60 ± 0.00
Lawyers	**0.60 ± 0.04**	0.16 ± 0.03	0.44 ± 0.14	0.36 ± 0.11	0.45 ± 0.03
Parliament	0.48 ± 0.02	0.01 ± 0.0	0.41 ± 0.05	0.40 ± 0.07	**0.51 ± 0.04**
COSN	0.91 ± 0.00	-0.0 ± 0.00	**1.00 ± 0.00**	0.73 ± 0.09	0.74 ± 0.12
SinaNet	0.28 ± 0.01	0.00 ± 0.00	**0.34 ± 0.02**	0.08 ± 0.05	0.25 ± 0.00

KEFRIN, by wining three out of five cases, is the winner of the real-world competition. DMoN is the winner of the Lawyers data set. And GDCRinTAN wins the competition for the parliament data set. Although GDCRinTAN does not win all cases; however, the improvements obtained by this method, compared to GDCinAN, can be considered as evidence for the effectiveness of the data space transformation and the adopted regularization.

5.2 Comparison using synthetic data with categorical features

We reported the results of synthetic attributed networks with categorical features in Tables 6.

Table 6: Comparison using synthetic data sets with categorical features: the average and standard deviation of ARI using ten random seed initializations are reported. The best results are boldfaced.

Network size	p, q, ζ	EVA	DMoN	KEFRiN	GDCinAN	GDCRinTAN
Small	0.9, 0.3, 0.9	0.185 ± 0.046	0.709± 0.101	**0.922 ± 0.119**	0.836 ± 0.160	0.849 ± 0.067
	0.9, 0.3, 0.7	0.211 ± 0.053	0.380 ± 0.107	0.819 ± 0.142	**0.885 ± 0.118**	0.541 ± 0.185
	0.9, 0.6, 0.9	0.266 ± 0.080	0.412 ± 0.109	0.726 ± 0.097	0.232 ± 0.044	**0.754 ± 0.057**
	0.9, 0.6, 0.7	0.321 ± 0.060	0.213 ± 0.051	**0.711 ± 0.145**	0.192 ± 0.048	0.314 ± 0.060
	0.7, 0.3, 0.9	0.126 ± 0.039	0.566 ± 0.105	**0.877 ± 0.130**	0.461 ± 0.100	0.735 ± 0.066
	0.7, 0.3, 0.7	0.126 ± 0.025	0.292 ± 0.077	**0.795 ± 0.117**	0.464 ± 0.096	0.308 ± 0.069
	0.7, 0.6, 0.9	0.015 ± 0.015	0.345 ± 0.064	**0.834 ± 0.132**	0.031 ± 0.010	0.740 ± 0.056
	0.7, 0.6, 0.7	0.008 ± 0.007	0.115 ± 0.058	**0.540 ± 0.107**	0.024 ± 0.01	0.306 ± 0.071
Medium	0.9, 0.3, 0.9	0.121 ± 0.031	0.512 ± 0.137	0.724 ± 0.097	0.388 ± 0.210	**0.842 ± 0.022**
	0.9, 0.3, 0.7	0.076 ± 0.038	0.272 ± 0.073	**0.742 ± 0.182**	0.475 ± 0.112	0.633 ± 0.110
	0.9, 0.6, 0.9	0.159 ± 0.046	0.370 ± 0.063	0.652 ± 0.110	0.001 ± 0.001	**0.837 ± 0.033**
	0.9, 0.6, 0.7	0.109 ± 0.046	0.168 ± 0.030	**0.733 ± 0.083**	0.002 ± 0.001	0.430 ± 0.020
	0.7, 0.3, 0.9	0.078 ± 0.036	0.446 ± 0.099	0.641 ± 0.111	0.007 ± 0.014	**0.814 ± 0.015**
	0.7, 0.3, 0.7	0.059 ± 0.010	0.228 ± 0.077	**0.797± 0.088**	0.002 ± 0.004	0.443 ± 0.041
	0.7, 0.6, 0.9	0.002 ± 0.002	0.332 ± 0.051	0.591 ± 0.094	0.001 ± 0.001	**0.800 ± 0.031**
	0.7, 0.6, 0.7	0.002 ± 0.002	0.133 ± 0.016	**0.773 ± 0.070**	0.001 ± 0.000	0.395 ± 0.025

In the small-size synthetic data competition KEFRIN is the winner and the proposed methods of this work and specially GDCRinTAN are its close followers. In the medium-size data sets, KEFRIN and GDCRinTAN both win four cases and can be considered as the winner of this competition, while the rest of the methods under consideration performed rather poorly. The improvement obtained by GDCRinTAN in row 3 and row 4 compared to GDCinAN at those settings, can be considered as evidence for the effectiveness of the proposed transformation and regularization.

5.3 Comparison over synthetic data with mixed scale features

We reported the results of synthetic data with mixed scale features in Tables 7.

Table 7: Comparison using synthetic data with mixed scale features: the average and standard deviation of ARI using ten random seed initializations are reported. The best results are boldfaced.

	small-size			medium-size		
p, q, ζ	KEFRiN	GDCinAN	GDCRinTAN	KEFRiN	GDCinAN	GDCRinTAN
0.9, 0.3, 0.9	0.752 ± 0.096	0.781 ± 0.118	**0.808 ± 0.054**	**0.834 ± 0.044**	0.415 ± 0.210	0.786 ± 0.013
0.9, 0.3, 0.7	0.769 ± 0.101	**0.803 ± 0.110**	0.636 ± 0.161	**0.801 ± 0.051**	0.385 ± 0.126	0.524 ± 0.028
0.9, 0.6, 0.9	**0.809 ± 0.138**	0.225 ± 0.048	0.768 ± 0.057	0.747 ± 0.071	-0.000 ± 0.004	**0.770 ± 0.024**
0.9, 0.6, 0.7	**0.716 ± 0.122**	0.175 ± 0.046	0.508 ± 0.071	**0.722 ± 0.059**	0.001 ± 0.001	0.492 ± 0.039
0.7, 0.3, 0.9	**0.750 ± 0.078**	0.542 ± 0.105	0.746 ± 0.058	**0.853 ± 0.048**	0.012 ± 0.029	0.763 ± 0.011
0.7, 0.3, 0.7	**0.681 ± 0.078**	0.458 ± 0.072	0.474 ± 0.040	**0.773 ± 0.060**	0.050 ± 0.050	0.518 ± 0.024
0.7, 0.6, 0.9	0.704 ± 0.139	0.028 ± 0.013	**0.736 ± 0.067**	0.726 ± 0.058	0.005 ± 0.004	**0.755 ± 0.032**
0.7, 0.6, 0.7	**0.540 ± 0.135**	0.028 ± 0.012	0.475 ± 0.087	**0.608 ± 0.037**	0.003 ± 0.002	0.499 ± 0.038

We observe similar patterns to the synthetic data sets with categorical attributes, that is, KEFRIN is the winner of small-size network and GDCRinTAN is its close follower. Similarly, in the medium-size networks we observe better performance of GDCRinTAN and the competitiveness of its obtained results.

6 Conclusion and future work

The current research reports our intermediate results towards developing an effective clustering method to recover communities in attributed networks. To achieve this objective, we relied on the data-driven model proposed in [8], and adopted the gradient descent optimization strategy, similar to what was done in [6], to optimized our clustering objective function. We called this version of our proposed method Gradient Descent Clustering in Attributed Network (GDCinAN). However, this straightforward method did not lead to promising results. To improve the quality of our results, we made two modifications: (a) we transformed the data spaces, using a softmax-scaled-dot-product, to a more cluster friendly data spaces; (b) we integrated the regularization term into our clustering objective function. We called this version as Gradient Descent Clustering with Regularization in Transformed Attributed Network (GDCRinTAN). With these two modifications we significantly improved the quality of the recovered clusters, compared to GDCinAN.

We evaluated and compared the performance of our proposed methods with three SOTA algorithms using five real-world and 320 synthetic networks. Our results showed that although GDCRinTAN was not the sole-winner, still it obtained comparable results and more so, in various experimental settings it outperformed the competitors. The current research is not without limits and those limits shape the directions of our future research.

1. adopting the L1 norm or elastic-net regularization,
2. adopting other distance functions, such as cosine distance function,
3. separating the regularization and the step size per each data source,
4. adopting different update rules,
5. proposing more sophisticated data-driven models to incorporated the topology of data, etc.

Acknowledgments Support from the Basic Research Program of HSE University is gratefully acknowledged. This research was supported in part through computational resources of HPC facilities at HSE University.

References

1. P. Chunaev, "Community detection in node-attributed social networks: a survey," *Computer Science Review*, vol. 37, p. 100286, 2020.
2. M. N. Al-Andoli, S. C. Tan, W. P. Cheah, and S. Y. Tan, "A review on community detection in large complex networks from conventional to deep learning methods: A call for the use of parallel meta-heuristic algorithms," *IEEE Access*, vol. 9, pp. 96 501–96 527, 2021.

3. M. Rostami, M. Oussalah, K. Berahmand, and V. Farrahi, "Community detection algorithms in healthcare applications: A systematic review," *IEEE Access*, 2023.
4. J. Cai, J. Hao, H. Yang, Y. Yang, X. Zhao, Y. Xun, and D. Zhang, "A new community detection method for simplified networks by combining structure and attribute information," *Expert Systems with Applications*, vol. 246, p. 123103, 2024.
5. J. Morand, S. Yip, Y. Velegrakis, G. Lattanzi, R. Potestio, and L. Tubiana, "Quality assessment and community detection methods for anonymized mobility data in the italian covid context," *Scientific Reports*, vol. 14, no. 1, p. 4636, 2024.
6. S. Shalileh, "An effective partitional crisp clustering method using gradient descent approach," *Mathematics*, vol. 11, no. 12, p. 2617, 2023.
7. S. Shalileh and B. Mirkin, "Community detection in feature-rich networks using gradient descent approach," in *International Conference on Complex Networks and Their Applications*. Springer, 2023, pp. 185–196.
8. S. Shalileh and Mirkin, "Summable and nonsummable data-driven models for community detection in feature-rich networks," *Social Network Analysis & Mining*, vol. 11, pp. 1–23, 2021.
9. D. R. Wilson and T. R. Martinez, "The general inefficiency of batch training for gradient descent learning," *Neural networks*, vol. 16, no. 10, pp. 1429–1451, 2003.
10. J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018. [Online]. Available: http://github.com/google/jax
11. E. Müller, "Graph clustering with graph neural networks," *Journal of Machine Learning Research*, vol. 24, pp. 1–21, 2023.
12. S. Citraro and G. Rossetti, "Identifying and exploiting homogeneous communities in labeled networks," *Applied Network Science*, vol. 5, no. 1, pp. 1–20, 2020.
13. S. Shalileh and B. Mirkin, "Community partitioning over feature-rich networks using an extended k-means method," *Entropy*, vol. 24, no. 5, p. 626, 2022.
14. L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification*, vol. 2, pp. 193–218, 1985.
15. D. Larremore, A. Clauset, and C. B. A, "network approach to analyzing highly recombinant malaria parasite genes," *PLoS Computational Biology*, vol. 9, no. 10, p. p.e1003268, 2013.
16. E. Lazega, *The Collegial Phenomenon: The Social Mechanisms of Cooperation Among Peers in a Corporate Law Partnership*, 1st ed. GB: Oxford University Press, 2001.
17. T. Snijders. (2001) Lawyers data set. Siena. [Online]. Available: https://www.stats.ox.ac.uk/~snijders/siena/
18. A. Bojchevski and S. Günnemanz, "Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure," in *Thirty-Second AAAI Conference on Artificial Intelligence*. California, USA: AAAI Press, 2018, pp. 1–10.
19. R. Cross and A. Parker, *The hidden power of social networks: Understanding how work really gets done in organizations*, 1st ed. USA: Harvard Business Press, 2004.
20. C. Jia, Y. Li, M. Carson, X. Wang, and J. Yu, "Node attribute-enhanced community detection in complex networks," *Scientific Reports*, vol. 7, no. 1, pp. 1–15, 2017.
21. S. Shalileh and B. Mirkin, "Least-squares community extraction in feature-rich networks using similarity data," *Plos one*, vol. 16, no. 7, p. e0254377, 2021.