Graph Encoder Ensemble for Simultaneous Vertex Embedding and Community Detection

CENCHENG SHEN, Department of Applied Economics and Statistics, University of Delaware, USA

YOUNGSER PARK, Department of Applied Mathematics and Statistics, Johns Hopkins University, USA

CAREY E. PRIEBE, Department of Applied Mathematics and Statistics, Johns Hopkins University, USA

In this paper, we introduce a novel and computationally efficient method for vertex embedding, community detection, and community size determination. Our approach leverages a normalized one-hot graph encoder and a rank-based cluster size measure. Through extensive simulations, we demonstrate the excellent numerical performance of our proposed graph encoder ensemble algorithm.

CCS Concepts: • **Computing methodologies** → *Ensemble methods*; *Cluster analysis*; • **Mathematics of computing** → *Graph algorithms*; *Nonparametric statistics*.

Additional Key Words and Phrases: Graph Embedding, Embedding Normalization, Vertex Clustering, Cluster Size Determination

1 INTRODUCTION

Graph data represents pairwise relationships between vertices through a collection of vertices and edges. Typically, a graph (or network) is represented by an adjacency matrix $\mathbf{A}$ of size $n \times n$, where $\mathbf{A}(i, j)$ denotes the edge weight between the ith and jth vertices. Alternatively, the graph can be stored in an edgelist $\mathbf{E}$ of size $s \times 3$, with the first two columns indicating the vertex indices of each edge and the last column representing the edge weight.

Community detection, also known as vertex clustering or graph partitioning, is a fundamental problem in graph analysis [6, 8, 10, 13]. The primary objective is to identify natural groups of vertices where intra-group connections are stronger than inter-group connections. Over the years, various approaches have been proposed, including modularity-based methods [2, 22], spectral-based methods [15, 21], and likelihood-based techniques [1, 7], among others.

Spectral-based and likelihood-based methods are extensively studied in the statistics community, but they tend to be computationally slow for large graphs. On the other hand, modularity-based methods are faster and widely used in practice, but they lack theoretical investigations and only provide community labels without vertex embedding. Moreover, determining the appropriate community size poses a challenge for any method and is often addressed in an ad-hoc manner or assumed to be known. Therefore, a desirable approach is to develop a method that can achieve community detection, vertex representation, and community size determination under a unified framework.

In this paper, we propose a graph encoder ensemble algorithm that simultaneously fulfills all these objectives. Our algorithm leverages a normalized one-hot graph encoder [19], ensemble learning [3, 12], k-means clustering [5, 11], and a novel rank-based cluster size measure called the minimal rank index. The proposed algorithm exhibits linear running time and demonstrates excellent numerical performance. The code for the algorithm is available on GitHub[1].

2 METHODS

We begin by introducing the one-hot graph encoder embedding from [19], known for its computational efficiency and theoretical guarantees under random graph models. This embedding forms the foundation of our proposed ensemble method, outlined in Algorithm 1. The ensemble algorithm incorporates crucial enhancements, including $L2$ normalization, the minimal rank index, and ensemble embedding, which are elaborated in the subsequent subsections.

2.1 Prerequisite

Given the graph adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ and a label vector $\mathbf{Y} \in 1, \ldots, K^n$, we define n_k as the number of observations per class, where

$$n_k = \sum_{i=1}^{n} 1(\mathbf{Y}_i = k)$$

for $k = 1, \ldots, K$. We construct the one-hot encoding matrix $\mathbf{W} \in \mathbb{R}^{n \times K}$ on $\mathbf{Y}$, then normalize it by the number of observations per-class. Specifically, for each vertex $i = 1, \ldots, n$, we set

$$\mathbf{W}(i, k) = \frac{1}{n_k}$$

if and only if $\mathbf{Y}_i = k$, and 0 otherwise. The graph encoder embedding is then obtained by performing a simple matrix multiplication:

$$\mathbf{Z} = \mathbf{AW} \in \mathbb{R}^{n \times K}.$$

Each row $\mathbf{Z}(i, :)$ represents a K-dimensional Euclidean representation of vertex i. The computational advantage of the graph encoder embedding lies in the matrix multiplications, which can be efficiently implemented by iterating over the edge list $\mathbf{E}$ only once, without the need for the adjacency matrix [19]. In Algorithm 1, we denote the above steps as

$$\mathbf{Z} = \text{one-hot-emb}(\mathbf{E}, \mathbf{Y}).$$

2.2 Main Algorithm

The proposed ensemble method is described in detail in Algorithm 1. It can be applied to binary or weighted graphs, as well as directed or undirected graphs. Throughout this paper, we set the number of random replicates $r = 10$, the maximum number of iterations $m = 20$, and the clustering range is determined based on the specific experiment.

In the pseudo-code, the $L2$ normalization step is represented by $\mathbf{Z} = \text{normalize}(\mathbf{Z})$, which normalizes each vertex representation to have unit norm (see Section 2.3 for more details). Additionally, given an embedding $\mathbf{Z}$ and a label vector $\mathbf{Y}$, the minimal rank index is denoted as $\text{MRI}(\mathbf{Z}, \mathbf{Y}) \in [0, 1]$, which measures the quality of clustering with a lower value indicating better clustering (details in Section 2.4). The k-means clustering step is denoted as k-means$(\mathbf{Z}, K)$, and the adjusted Rand index is denoted as $\text{ARI}(\mathbf{Y}, \mathbf{Y}_2)$, which measures the similarity between two label vectors of the same

---

[1]https://github.com/cshen6/GraphEmd

size. The ARI is a popular matching metric that ranges from $-\infty$ to 1, with a larger positive value indicating better match quality and a value of 1 representing a perfect match [14].

Algorithm 1 Graph Encoder Ensemble

Require: An edgelist $\mathbf{E}$, a range of potential cluster size R, number of random replicates r, and number of maximum iteration m.

Ensure: The graph embedding $\mathbf{Z} \in \mathbb{R}^{n \times \hat{K}}$ for all vertices, the estimated number of clusters $\hat{K}$, the cluster indices $\mathbf{Y} \in \mathbb{R}^n$, and the minimal rank index $ind \in [0, 1]$.

   function Graph-Encoder-Ensemble($\mathbf{E}, R, r, m$)

      $ind = 1$;　　　　　　　　　　　　　　　　　　　▷ initialize the index to pick best cluster size

      for $k \in R$ **do**

         $ind_2 = 1$;　　　　　　　　　　　　　　　　　▷ initialize the index to pick best random replicate

         for $i = 1, \ldots, r$ **do**

            $\hat{\mathbf{Y}}_k = rand(k, n)$;　　　　　　　　　　　　▷ randomly initialize a label vector of length n in $[k]$

            for $j = 1, \ldots, m$ **do**

               $\hat{\mathbf{Z}}_k = $ one-hot-emb($\mathbf{E}, \hat{\mathbf{Y}}_k$);

               $\hat{\mathbf{Z}}_k = $ normalize($\hat{\mathbf{Z}}_k$);

               $\hat{\mathbf{Y}}'_k = $ k-means($\hat{\mathbf{Z}}_k, k$);

               if ARI($\hat{\mathbf{Y}}_k, \hat{\mathbf{Y}}'_k$)==1 **then** break;

               else $\hat{\mathbf{Y}}_k = \hat{\mathbf{Y}}'_k$;

               end if

            end for

            $\hat{\mathbf{Z}}_k = $ one-hot-emb($\mathbf{E}, \hat{\mathbf{Y}}_k$);

            $\hat{\mathbf{Z}}_k = $ normalize($\hat{\mathbf{Z}}_k$);

            $ind_3 = $ MRI($[\hat{\mathbf{Z}}_k, \hat{\mathbf{Y}}_k]$);

            if $ind_3 < ind_2$ **then**

               $\hat{\mathbf{Z}} = \hat{\mathbf{Z}}_k; \hat{\mathbf{Y}} = \hat{\mathbf{Y}}_k; ind_2 = ind_3$;

            end if

         end for

         if $ind_2 \leq ind$ **then**

            $\mathbf{Z} = \hat{\mathbf{Z}}; \mathbf{Y} = \hat{\mathbf{Y}}; \hat{K} = k; ind = ind_2$;

         end if

      end for

   end function

2.3 Why Normalization

The normalization step in Algorithm 1 scales each vertex embedding to unit norm. Specifically, for each vertex i,

$$\mathbf{Z}(i, :) = \mathbf{Z}(i, :) / \|\mathbf{Z}(i, :)\|_2.$$

if $\|\mathbf{Z}(i, :)\|2 > 0$. The normalization step plays a crucial role in achieving improved clustering results, as demonstrated in Figure 1 using a sparse random graph model with two communities. The normalized embedding is represented on

a unit sphere, effectively capturing the connectivity information while mitigating the influence of vertex degrees. In contrast, the un-normalized embedding is significantly affected by the original vertex degrees, resulting in vertices from the same community being widely dispersed. This distinction bears resemblance to the two-truth phenomenon observed in graph adjacency and graph Laplacian, where the Laplacian spectral embedding (LSE) can be seen as a degree-normalized version of the adjacency spectral embedding (ASE). The LSE typically performs better on sparse graphs. Further numerical evaluations on the normalization effect can be found in Section 3.2 and Table 1.
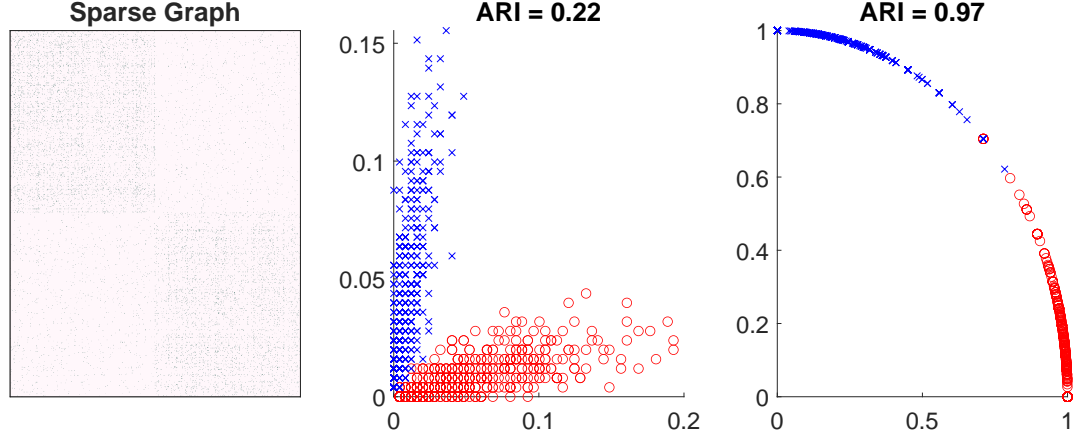

Fig. 1. This figure visually demonstrates the effect of normalization. The left panel displays the adjacency heatmap of a simulated sparse graph using simulation 1 in Section 3.1. The center panel shows the resulting embedding without the normalization step, while the right panel displays the resulting embedding with normalization. The blue and red dots represent the true community labels of each vertex.

2.4 The Minimal Rank Index

We introduce a new rank-based measure called the minimal rank index (MRI) to assess the quality of clustering. This measure plays a crucial role in Algorithm 1 as it enables the comparison of multiple embeddings generated from different initializations and community sizes.

Given the cluster index $\mathbf{Y}i$ of vertex i, the Euclidean distance function $d(\cdot, \cdot)$, and the mean of the kth cluster denoted as

$$\mu_k = \sum i = 1, \ldots, n, \mathbf{Y}i = k\mathbf{Z}i,$$

the minimal rank index is computed as:

$$MRI = \sum_{i=1,\ldots,n} I\{\arg \min_{k=1,\ldots,K} d(\mathbf{Z}_i, \mu_k) \neq \mathbf{Y}_i\}/n \in [0, 1]. \tag{1}$$

The MRI measures how often the vertex embedding is not closest to its corresponding cluster mean. A smaller MRI value indicates better clustering quality, with MRI equal to 0 indicating that every vertex is closest to its cluster mean. In the context of k-means clustering, MRI is non-zero when the k-means algorithm fails to converge.

In comparison to common cluster size measures such as Silhouette Score, Davies-Bouldin index, Variance Ratio Criterion, and Gap criterion [4, 16], MRI is rank-based rather than based on actual distances. These other measures

4

compute ratios of within-cluster distances to between-cluster distances. If any of these measures were used in Algorithm 1 instead of MRI, the choice of cluster size would be biased towards the smallest possible value. This is due to the incremental nature of graph encoder embedding in Algorithm 1, where the embedding dimension is equal to the community size k. Consequently, within-cluster distances become smaller for smaller values of k, resulting in a bias towards the smallest k when using actual distance.

2.5 Ensemble Embedding and Cluster Size Determination

Ensemble learning is utilized in Algorithm 1 to improve learning performance and reduce variance by employing multiple models. The approach can be summarized as follows: for each value of k in the cluster range, we generate a set of vertex embeddings and community labels using random label initialization. The model with the smallest MRI is selected as the best model. In cases where multiple models have the same smallest MRI, the average embedding is used.

   Additionally, among all possible choices of cluster size k, the best embedding with the smallest MRI is selected. If there are multiple embeddings with the same smallest MRI, the one with the largest k is chosen. For instance, if the MRI values are $0, 0, 0, 0.1, 0.2$ for $K = 2, 3, 4, 5, 6$, the graph encoder ensemble would select $\hat{K} = 4$.

2.6 Computational Complexity Analysis

Algorithm 1 comprises several steps, including one-hot graph encoder embedding, k-means clustering, MRI computation, and ensembles. Let n be the number of vertices and s be the number of edges. At any fixed k, the one-hot graph encoder embedding takes $O(nk + s)$, k-means takes $O(nk)$, and the MRI computation takes $O(nk)$. Therefore, the overall time complexity of Algorithm 1 is $O(rm(n \max(R) + s))$, which is linear with respect to the number of vertices and edges. The storage requirement is also $O(n \max(R) + s)$. In practical terms, the graph encoder ensemble algorithm exhibits remarkable efficiency and scalability. Testing on simulated graphs with default parameters and $\max(R) = 10$, it takes less than 3 minutes to process 1 million edges and less than 20 minutes for 10 million edges.

3 RESULTS

In this section, we conduct extensive numerical experiments to demonstrate the advantages of the graph encoder ensemble, as well as the individual benefits of normalization, ensemble, and MRI. We compare these approaches against benchmarks including the algorithm without normalization, without ensemble, with MRI replaced, and using adjacency/Laplacian spectral embedding. The performance is evaluated using the adjusted Rand index (ARI), which measures the degree of agreement between the estimated communities and the ground-truth labels.

3.1 Simulation Set-up

The stochastic block model (SBM) is a widely used random graph model for studying community structure [9, 20]. Each vertex i is associated with a class label $Y_i \in \{1, \ldots, K\}$. The class label may be fixed a-priori, or generated by a categorical distribution with prior probability $\{\pi_k \in (0, 1)$ with $\sum_{k=1}^{K} \pi_k = 1\}$. Then a block probability matrix $\mathbf{B} = [\mathbf{B}(k, l)] \in [0, 1]^{K \times K}$ specifies the edge probability between a vertex from class k and a vertex from class l. For any $i < j$,

$$\mathbf{A}(i, j) \overset{i.i.d.}{\sim} \text{Bernoulli}(\mathbf{B}(Y_i, Y_j)),$$

$$\mathbf{A}(i, i) = 0, \quad \mathbf{A}(j, i) = \mathbf{A}(i, j).$$

The degree-corrected stochastic block model (DC-SBM) [23] is a generalization of SBM to better model the sparsity of real graphs. Everything else being the same as SBM, each vertex i has an additional degree parameter θ_i, and the adjacency matrix is generated by

$$\mathbf{A}(i, j) \sim \text{Bernoulli}(\theta_i \theta_j \mathbf{B}(Y_i, Y_j)).$$

In our simulations, we consider three DC-SBM models with increasing community sizes. In all models, the degrees are generated randomly by $\theta_i \overset{i.i.d.}{\sim} Beta(1, 4)$.

Simulation 1: $n = 3000$, $K = 2$, $Y_i = \{1, 2\}$ equally likely, and the block probability matrix is

$$\mathbf{B} = \begin{bmatrix} 0.5, 0.1 \\ 0.1, 0.5 \end{bmatrix}.$$

Simulation 2: $n = 3000$, $K = 4$, $Y_i = \{1, 2, 3, 4\}$ with prior probability $[0.2, 0.2, 0.3, 0.3]$, and the block probability matrix is

$$\mathbf{B} = \begin{bmatrix} 0.9, 0.1, 0.1, 0.1 \\ 0.1, 0.7, 0.1, 0.1 \\ 0.1, 0.1, 0.5, 0.1 \\ 0.1, 0.1, 0.1, 0.3 \end{bmatrix}.$$

Simulation 3: $n = 3000$, $K = 5$, Y_i with equally likely prior probability, and the block probability matrix satisfies $\mathbf{B}(i, i) = 0.2$ and $\mathbf{B}(i, j) = 0.1$ for all $i = 1, \ldots, 5$ and $j \neq i$.

3.2 Normalization Comparison

Table 1 provides clear evidence of the superior clustering performance achieved by the normalized algorithm compared to the un-normalized algorithm. To isolate the impact of normalization, we set $r = 1$ and assume the cluster size is known. The observed improvement aligns with the phenomenon observed between adjacency spectral embedding (ASE) and Laplacian spectral embedding (LSE), where LSE, being a normalized version of ASE, consistently outperforms ASE.

	ARI			
	GEE	GEE no norm	ASE	LSE
Simulation 1	**0.91**	0.10	0.23	**0.91**
Simulation 2	**0.73**	0.08	0.12	0.65
Simulation 3	**0.78**	0.06	0.17	**0.78**

Table 1. This table demonstrates the advantage of normalization in the graph encoder ensemble. The "GEE" column refers to the graph encoder ensemble using Algorithm 1, while "GEE no norm" indicates that normalization is not applied. The reported results are averages obtained from 100 Monte Carlo replicates.

3.3 Ensemble Comparison

In this simulation, we assume a known cluster size and conduct 100 Monte Carlo replicates to compare the performance of the ensemble algorithm ($r = 10$) with the no-ensemble version ($r = 1$). The results in Table 2 clearly demonstrate the

superiority of the ensemble algorithm: it achieves higher mean ARI and significantly reduces the variance compared to the no-ensemble version. Based on our empirical observations, the default choice of $r = 10$ yields satisfactory results across our experiments. Additionally, if the graph size is sufficiently large and the community structure is well-separated, using a smaller value of r or even $r = 1$ is sufficient. This is evident in simulation 1 of Table 2.

	Average ARI + std	
	GEE	GEE ($r = 1$)
Simulation 1	0.91 ± 0.01	0.91 ± 0.01
Simulation 2	0.79 ± 0.02	0.72 ± 0.09
Simulation 3	0.89 ± 0.01	0.79 ± 0.12

Table 2. This table assesses the advantage of the ensemble approach in the graph encoder ensemble. The reported results include the mean and standard deviation of the Adjusted Rand Index (ARI) obtained from 100 Monte Carlo replicates.

3.4 Cluster Size Estimation

In this analysis, we explore the performance of the algorithm in estimating the community size. Instead of using the ground-truth size, we consider a range of potential sizes from $R = 2$ to $R = 10$, and the results are presented in Figure 2. These findings provide insights into the performance of the algorithm in accurately estimating the community size and highlight the importance of the MRI measure in achieving accurate size determination.
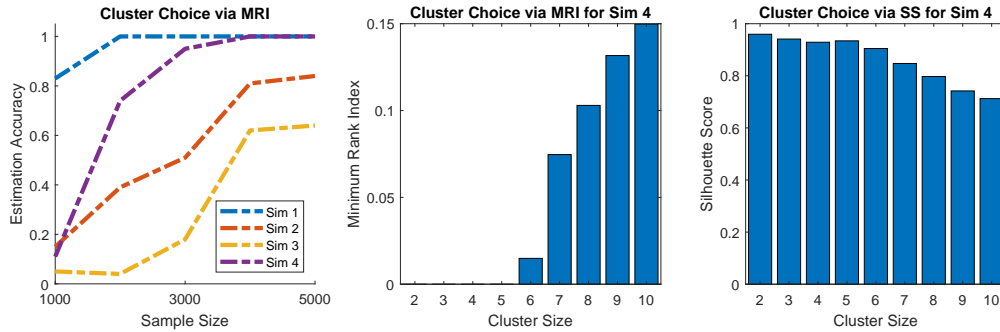

Fig. 2. This figure presents the results of cluster size estimation using the graph encoder ensemble. The estimation accuracy and the performance of different size measures are evaluated for various simulations and graph sizes. For each simulation and each graph size, we independently generate 100 graphs, and run the ensemble algorithm to estimate the community size. The left panel of the figure illustrates the estimation accuracy as the graph size increases. The estimation accuracy represents the proportion of cases where the algorithm correctly chooses the community size. As the graph size increases, the estimation accuracy gradually improves, reaching a perfect estimation accuracy of 1 for all simulations. The center panel focuses on simulation 3 at $n = 5000$. The MRI calculates $\hat{K} = 5$ as the estimated community size, which matches the ground-truth size. In the right panel, the average Silhouette Score is computed as an alternative size measure, which is biased towards smaller community sizes and chooses $\hat{K}_{SS} = 2$, resulting in a different estimation compared to the ground-truth size.

4 CONCLUSION

This paper introduces the graph encoder ensemble, which achieves graph embedding, community detection, and community size determination in a unified framework. Its main advantages include ease of implementation, computational efficiency, and excellent performance in community detection and community size selection. Several potential future directions include exploring mathematical proofs for asymptotic clustering optimality, investigating theoretical properties of MRI, and extending the method to dynamic and multi-modal graphs [17, 18].

REFERENCES

[1] Emmanuel Abbe. 2018. Community Detection and Stochastic Block Models: Recent Developments. *Journal of Machine Learning Research* 18, 177 (2018), 1–86.

[2] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 10008 (2008), 6.

[3] L. Breiman. 2001. Random Forests. *Machine Learning* 4, 1 (October 2001), 5–32.

[4] David L. Davies and Donald W. Bouldin. 1989. A Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1, 2 (1989), 224–227.

[5] Edward W. Forgy. 1965. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics* 21, 3 (1965), 768–769.

[6] Santo Fortunato. 2010. Community detection in graphs. *Physics Reports* 486, 3–5 (2010), 75–174. https://doi.org/10.1016/j.physrep.2009.11.002 arXiv:arXiv:0906.0612

[7] Chao Gao, Zongming Ma, Anderson Y. Zhang, and Harrison H. Zhou. 2018. Community detection in degree-corrected block models. *Annals of Statistics* 46, 5 (2018), 2153–2185.

[8] M. Girvan and M. E. J. Newman. 2002. Community Structure in Social and Biological Networks. *Proceedings of National Academy of Science* 99, 12 (2002), 7821–7826.

[9] P. Holland, K. Laskey, and S. Leinhardt. 1983. Stochastic Blockmodels: First Steps. *Social Networks* 5, 2 (1983), 109–137.

[10] B. Karrer and M. E. J. Newman. 2011. Stochastic blockmodels and community structure in networks. *Physical Review E* 83 (2011), 016107.

[11] Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory* 28, 2 (1982), 129–137.

[12] R. Maclin and D. Opitz. 1999. Popular Ensemble Methods: An Empirical Study. *Journal Of Artificial Intelligence Research* 11 (1999), 169–198.

[13] M. E. J. Newman. 2004. Detecting community structure in networks. *European Physical Journal B* 38, 2 (2004), 321–330. https://doi.org/10.1140/epjb/e2004-00124-y

[14] W. M. Rand. 1971. Objective criteria for the evaluation of clustering methods. *J. Amer. Statist. Assoc.* 66, 336 (1971), 846–850.

[15] K. Rohe, S. Chatterjee, and B. Yu. 2011. Spectral Clustering and the High-Dimensional Stochastic Blockmodel. *Annals of Statistics* 39, 4 (2011), 1878–1915.

[16] Peter J. Rousseeuw. 1987. Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis. *Computational and Applied Mathematics* 20 (1987), 53–65.

[17] C. Shen, J. Larson, H. Trinh, X. Qin, Y. Park, and C. E. Priebe. 2023. Discovering Communication Pattern Shifts in Large-Scale Labeled Networks using Encoder Embedding and Vertex Dynamics. *https://arxiv.org/abs/2305.02381* (2023).

[18] C. Shen, C. E. Priebe, J. Larson, and H. Trinh. 2023. Synergistic Graph Fusion via Encoder Embedding. *https://arxiv.org/abs/2303.18051* (2023).

[19] C. Shen, Q. Wang, and C. E. Priebe. 2023. One-Hot Graph Encoder Embedding. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 6 (2023), 7933 – 7938.

[20] T. Snijders and K. Nowicki. 1997. Estimation and Prediction for Stochastic Blockmodels for Graphs with Latent Block Structure. *Journal of Classification* 14, 1 (1997), 75–100.

[21] D. Sussman, M. Tang, D. Fishkind, and C. Priebe. 2012. A Consistent Adjacency Spectral Embedding for Stochastic Blockmodel Graphs. *J. Amer. Statist. Assoc.* 107, 499 (2012), 1119–1128.

[22] V. A. Traag, L. Waltman, and N. J. van Eck. 2019. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports* 9 (2019), 5233.

[23] Y. Zhao, E. Levina, and J. Zhu. 2012. Consistency of Community Detection in Networks under Degree-Corrected Stochastic Block Models. *Annals of Statistics* 40, 4 (2012), 2266–2292.

ACKNOWLEDGMENTS