Graph Vertex Embeddings: Distance, Regularization and Community Detection

Radosław Nowak[1 2][0009-0009-2649-177X], Adam Małkowski[0000-0001-8304-6837], Daniel Cieślak[1 3][0009-0001-6046-5315], Piotr Sokół[1][0009-0000-6889-7641], and Paweł Wawrzyński[1][0000-0002-1154-0470]

[1]IDEAS NCBR, https://ideas-ncbr.pl, [2]Polish Academy of Sciences, https://pan.pl/en/, [3]Gdańsk University of Technology, https://pg.edu.pl/en

Abstract. Graph embeddings have emerged as a powerful tool for representing complex network structures in a low-dimensional space, enabling efficient methods that employ the metric structure in the embedding space as a proxy for the topological structure of the data. In this paper, we explore several aspects that affect the quality of a vertex embedding of graph-structured data. To this effect, we first present a family of flexible distance functions that faithfully capture the topological distance between different vertices. Secondly, we analyze vertex embeddings as resulting from a fitted transformation of the distance matrix rather than as a direct result of optimization. Finally, we evaluate the effectiveness of our proposed embedding constructions by performing community detection on a host of benchmark datasets. The reported results are competitive with classical algorithms that operate on the entire graph while benefitting from a substantially reduced computational complexity due to the reduced dimensionality of the representations.

Keywords: Graphs · Embeddings · Graph drawing · Community detection.

1 Introduction

Low-dimensional metric embeddings of non-metric data play a crucial role in various domains of computer science, e.g.: **a**) machine learning, where probabilistic generative models are constructed to capture variability through low dimensional factors [53,26]; **b**) natural language processing, where symbolic/text data is represented vectorially in order to facilitate learning of statistical dependencies [50]; **c**) information retrieval, where embeddings allow for efficient search [6]; **d**) data visualization, where complex, high-dimensional data is represented in a two- or three-dimensional space [8]. Crucially, low-dimensional embeddings of data mitigate the computational and statistical challenges collectively referred to as the *curse of dimensionality*.

Graph embeddings present a particularly interesting potential application since the data is inherently non-metric and high-dimensional while simultaneously being of tremendous practical interest, due to the ubiquity of graphs in various

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63778-0_4

domains, e.g.: social networks, biological networks, energy grids, and knowledge graphs [24,38]. Addressing the challenges posed by 'graph problems' requires faithful and compact representations of the data, i.e. they should retain information about important structural properties and allow flexible, computationally efficient use for downstream processing/tasks.

Graph embeddings have been widely studied; past approaches include spectral embeddings, graph kernels, multi-dimensional scaling (MDS), locally linear embedding (LLE), and Laplacian eigenmaps (LE)[56,2]. Recently, graph neural networks (GNNs) have emerged as an empirically successful model class, which offers improved scalability and more flexible processing of the embeddings [56].

In this work, we introduce a novel method for representing graph-structured data in low-dimensional metric spaces, which combines the efficiency of optimization-based embeddings and the expressiveness of neural network-based approximate to accurately reflect the topological distances within the graph. By framing the embedding of vertices as an optimization problem, we parametrize the embedding using a small neural network, to regularize the resulting representation. Furthermore, our formulation can accommodate various distance functions, which allows us to adapt the geometry of the embedding space to better reflect the structure of the original graph. The resulting embeddings offer a compact yet faithful representation, which combined with off-the-shelf clustering algorithms allow us to effectively identify communities within the graph. Our approach not only ensures a more favorable computational and statistical scaling, comparable to that of Graph Neural Networks (GNNs) but also provides a highly expressive representation of the graph structure, capturing the intricate relationships and distances within the data.

2 Related work

The body of research on embeddings is extensive and has a long history. Consequently, we begin by recalling foundational results before providing a concise overview of the methodologies presently in use. For a comprehensive review see [56,60].

Historically, the graph embedding problem has been initially studied in the context of dimensionality reduction and data visualization, while preserving important structural properties of the data. Seminal examples of such methods include PCA, graph kernels [9], Laplacian eigenmaps [4], Isomap [49], LLE [43], maximum variance unfolding [55], t-SNE [32], LargeVis [48], UMAP [34], and the latent variable models (LVM).

[2] introduced *minimum distortion embeddings* as a unifying framework that subsumes all of the previously mentioned methods, except t-SNE and LVM. In their formulation, low-dimensional embeddings are constructed by minimizing the distortion of pairwise distances between data points in the original and the embedded spaces. Independently, [23,10,31,21] have shown that an m-dimensional

embedding of graph with n vertices incurs a distortion of order $\mathcal{O}(\log n)$, where m is $\mathcal{O}\left(\log^2 n\right)$. Remarkably, the landmark results of Johnson, Lindenstrauss, and Bourgain [31] not only provide the mathematical foundation of low-dimensional representations of data but also have led to the development of fast, randomized algorithms.

Currently prevalent algorithmic approaches present a variety of design options. An initial consideration requires the specification of a geometry for the embedding space; potential choices include hyperbolic [40,28,44], spherical [2], and vector embeddings. The latter are compatible with a wide range of machine-learning algorithms and have proven versatile in disparate domains [35,30], and model classes, e.g.: Node2vec [19]; graph neural networks [51,17]; Isomap [49]; M-NMF [54]. Among these, we distinguish graph neural networks (GNNs) [51,17,36] due to their wide-spread adoption, and ability to directly learn embeddings from graph-structured data. GNNs are predominantly trained in a supervised or semi-supervised fashion [27]; optimizing a node classification loss, potentially augmented by auxiliary terms, e.g. reconstruction error or noise contrastive estimate [52,11,13]. The resulting algorithms have proven to be efficient, and empirically successful when applied to tasks such as graph regression, node classification, and link prediction [15].

Comparatively, the use of GNNs for community detection has been relatively underexplored [46], despite the pivotal role that communities play in understanding the structure of biology, social, and economic networks [57]. In this context, the applicability of GNNs is curtailed by their reliance on labeled data, which is often unavailable or requires cost-intensive curation.

Classically, clustering algorithms have been used to address the community detection problem, e.g.: Girvan-Newman [18], Louvain [7], and spectral clustering [41] algorithms. More recent approaches, e.g. [46,22], use GNN's to construct an embedding of the data, which is then post-processed using an off-the-shelf clustering algorithm, such as mean shift [29], DBSCAN [16], HDBSCAN [33], Birch [59], OPTICS [3], AffinityPropagation [47], AgglomerativeClustering [39]. Our proposed approach improves on GNN approaches by offering a lightweight neural network model, that can be optimized efficiently without label supervision.

Recent studies citing Agrawal's work underscore its relevance across diverse biological research areas. For example, research highlighting the pitfalls of extreme dimensionality reduction in single-cell genomics– [12] leverages Agrawal's insights to critique conventional visualization techniques, advocating for targeted embedding strategies for more meaningful biological analysis. Similarly, an integrated single-cell dataset study of the hypothalamic paraventricular nucleus (PVN) [5] applies Agrawal's principles to achieve a nuanced molecular and functional classification, revealing the complexity of neuroendocrine regulation. Additionally, advancements in brain-wide neuronal activity recording through blazed oblique plane microscopy, as detailed in a study from Nature [20], demonstrate the utility of Agrawal's work in bridging cellular and macroscale understanding. This particular study challenges the prevailing view that higher resolution is invariably superior, illustrating the impact of Agrawal's contributions to enhancing our understanding of complex biological systems.

3 Problem statement

We consider a graph given by a set of vertices, $V = \{v_1, \ldots, v_n\}$, where $|V| = n$ is the order of the graph. Given a fixed, we represent the shortest path length between pairs of vertices as the matrix $D \in \mathbb{R}_+^{n\times n}$. These shortest distances may be based on unitary distances between adjacent vertices but also on any positive distances between adjacent vertices. In the present study, we focus exclusively on undirected graphs; therefore, D is symmetric. In the case of disconnected graphs, we assume the topological distance between vertices in different subgraphs is equal to the maximum distance between vertices in the same subgraph plus one.

The problem is to assign an embedding, $e_i \in \mathbb{R}^m$, to each vertex $v_i \in V$, in such a way that the approximate equality

$$d(e_i, e_j) \cong D_{i,j} \tag{1}$$

holds for each $i, j \in \{1, \ldots, n\}$, where $d(\cdot, \cdot)$ is a certain distance in $\mathbb{R}^m$, e.g., the Euclidean distance.

4 Method

4.1 Embeddings as solution to optimization problem

Following [2], we obtain the embeddings e_i (1) as a solution to an optimization problem which minimizes the average discrepancy between $d(e_i, e_j)$ and $D_{i,j}$:

$$[e_1, \ldots, e_n] = \operatorname{argmin}_{[e_1,\ldots,e_n]} \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{i-1} L(d(e_i, e_j), D_{i,j}). \tag{2}$$

These discrepancies, denoted above by $L(\cdot, \cdot)$, may be absolute

$$L(d(e_i, e_j), D_{i,j}) = (d(e_i, e_j) - D_{i,j})^2 \tag{3}$$

or relative

$$L(d(e_i, e_j), D_{i,j}) = (d(e_i, e_j)/D_{i,j} - 1)^2. \tag{4}$$

4.2 Regularized embeddings

We notice that we can identify a vertex by a vector of distances to all vertices in the graph, i.e., a row or column in the D matrix. This is because such a vector contains only one 0, and its position identifies the vertex of the question. Moreover, if two columns in the D matrix are similar, their respective vertices are in similar distance to other vertices and are usually close. Therefore, their respective embeddings should also be close. Therefore, we consider the embeddings as results of continuous transformations of the columns in the D matrix. This continuity is a form of regularization that prevents the optimization process (2) from getting stuck in a poor local minimum.

We consider the above transformation in the form of the deep neural network,

$$e_j = f(D_{\cdot,j}; \theta), \tag{5}$$

where f denotes the network and θ represents its weights. Then, to compute the embeddings, we need to optimize the weights θ of the network:

$$\theta = \text{argmin}_\theta \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{i-1} L(d(f(D_{\cdot,i}; \theta), f(D_{\cdot,j}; \theta)), D_{i,j}). \tag{6}$$

4.3 Distance functions

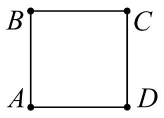

Fig. 1. Simple square graph

Let us consider the graph in Fig. 1 to motivate the need for flexible distance functions in the embedding space. The topological distance for adjacent vertices equals 1, and for diagonally-opposite vertices equals 2. Under an ℓ_2 distance, there is no possible arrangement of the embeddings e_i in $\mathbb{R}^m$ that preserves $D_{i,j}$. If the distance between the adjacent vertex embeddings equals 1, the distances between both pairs of opposite vertex embeddings cannot equal 2 *at the same time*.

Let us consider a generalized distance formula

$$d(e_i, e_j) = \|e_i - e_j\|^\kappa, \tag{7}$$

where $\kappa \in [0, 2]$. For $\kappa = 1$, (7) represents the Euclidean distance. Treating κ as a parameter, we optimize it jointly, along with the vertex embeddings of each graph.

Coming back to the graph in Fig. 1, let us consider $m = 2$ and the most natural embeddings for the vertices: $e_A = [0, 0]^T$, $e_B = [0, 1]^T$, $e_C = [1, 1]^T$, $e_D = [1, 0]^T$. It is seen that for $\kappa = 2$, we have $d(e_i, e_j) = 1$ for each pair of adjacent vertices and $d(e_i, e_j) = 2$ for each pair of the opposite vertices. This exactly reflects the topological distances.

For an arbitrary graph, variable κ may improve the fitness of embeddings measured by (3) and by (4). Additionally, optimizing κ to minimize the discrepancy allows the model to adapt to the specific structure of the graph, which may be beneficial for the quality of the embeddings.

5 Experiments

Our experiments are centered on three key areas, which we have identified as particularly compelling during the development of our method. These encompass evaluating the quality of graph embeddings determined by the loss function, generating graphs with embeddings produced by our method, and identifying communities within graphs using our approach. The challenge of community detection can be assessed through the *modularity* of the detected communities and by comparing them to ground truth communities. Since we did not restrict our benchmarks to sets of graphs exclusively comprised of community structures, most of our measurements are based on the *modularity* metric.

To verify the neural netowrk approach, we used a multilayer perceptron composed of four hidden layers with: 2048, 1024, 512, 256 neurons. Each hidden layer uses the ReLU activation function.

5.1 Datasets

For graph analysis and modularity measurement, we employed *TUDataset* [37], specifically opting for one graph set from each thematic category within the dataset. The selected graph sets are as follows: *MUTAG (small molecules)*, *ENZYMES (Bioinformatics)*, *Cuneiform (Computer vision)*, *IMDB-BINARY (Social networks)*, and *SYNTHETIC (Synthetic)*. These sets comprise graphs of relatively small size and similar characteristics.

Additionally, we utilized the *CORA* dataset [45], which features a single, large graph. For evaluating the community detection task, we employed two specific graphs representing real-world communities: the *Zachary Karate Club* [58] and *American Football* [18]. To visualize how different parameters affect final embeddings, we employed *American Revolution* graph [1].

5.2 Graph analysis

This section is dedicated to presenting the results regarding the performance of each analyzed method in preserving the actual topological distances between graph nodes within the embedding space. We introduce two metrics, RMSE and RMRSE (depending on the utilized loss function for generating embeddings), defined for a single graph as follows:

$$RMSE = \sqrt{\frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{i-1} (d(e_i, e_j) - D_{i,j})^2} \tag{8}$$

$$RMRSE = \sqrt{\frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{i-1} (d(e_i, e_j)/D_{i,j} - 1)^2}. \tag{9}$$

To analyze both the RMSE and RMRSE measures, we compared two approaches: a predefined $\kappa = 1$ with a value of 1.0 (Table 1) and an automatically learned κ (Table 2). From both tables, we observe that the loss generally decreases as the dimensionality of embeddings increases. This observation is expected, as embeddings in higher-dimensional vector spaces can compress a graph's nodes to preserve topological distances for more complex graphs. However, for each dataset, there is a specific limit embedding dimension above which its further increase does not lead to any loss gain.

As seen in Table 1, the embeddings optimized directly provide us with worse results than those produced by the fitted neural network. This is especially visible for small embedding dimensions. The regularization introduced by the neural network prevents the optimization of the embeddings from getting stuck in local minima. For larger dimensions, the neural network does not introduce any improvement. The only exception is the CORE graph, for which the network could not learn a well-fitting transformation.

Comparing Table 1 with Table 2, we note that the automatically learned κ parameter yields better results, usually by a large margin.

In the Table 3 we present values of the automatically learned κ parameter. We see the κ usually increases with m. This relation reflects the increasing difficulty in preserving the topological distances of vertices in embedding space of a decreasing dimension. Notably, for large m, the optimal κ is usually well above 1.

Method	Dataset	m=2	m=3	m=5	m=10	m=15	m=30	m=50
RMSE Direct	MUTAG	0.38 ± 0.23	0.24 ± 0.04	0.24 ± 0.03	0.24 ± 0.03	0.24 ± 0.03	0.24 ± 0.03	0.24 ± 0.03
	Cuneiform	0.58 ± 0.16	0.33 ± 0.13	0.17 ± 0.07	0.11 ± 0.02	0.11 ± 0.02	0.11 ± 0.02	0.11 ± 0.02
	SYNTHETIC	1.32 ± 0.04	0.94 ± 0.03	0.60 ± 0.01	0.44 ± 0.00	0.43 ± 0.00	0.43 ± 0.00	0.43 ± 0.00
	ENZYMES	0.53 ± 0.71	0.30 ± 0.44	0.22 ± 0.28	0.20 ± 0.15	0.20 ± 0.11	0.20 ± 0.10	0.19 ± 0.10
	IMDB-BINARY	0.38 ± 0.09	0.25 ± 0.06	0.15 ± 0.04	0.09 ± 0.03	0.07 ± 0.03	0.07 ± 0.04	0.07 ± 0.04
	CORA	3.30	2.47	1.80	1.17	0.90	0.57	0.45
RMSE Neural	MUTAG	0.29 ± 0.05	0.25 ± 0.04	0.25 ± 0.03	0.25 ± 0.04	0.25 ± 0.04	0.25 ± 0.03	0.25 ± 0.03
	Cuneiform	0.55 ± 0.14	0.33 ± 0.12	0.18 ± 0.07	0.12 ± 0.02	0.12 ± 0.02	0.12 ± 0.02	0.11 ± 0.02
	SYNTHETIC	1.20 ± 0.05	0.88 ± 0.03	0.60 ± 0.00	0.44 ± 0.00	0.44 ± 0.00	0.44 ± 0.00	0.44 ± 0.00
	ENZYMES	0.34 ± 0.22	0.26 ± 0.14	0.24 ± 0.11	0.23 ± 0.10	0.23 ± 0.10	0.23 ± 0.10	0.24 ± 0.12
	IMDB-BINARY	0.38 ± 0.09	0.26 ± 0.06	0.16 ± 0.04	0.09 ± 0.03	0.08 ± 0.04	0.08 ± 0.04	0.08 ± 0.04
	CORA	2.28	1.82	1.53	1.35	1.40	1.45	1.45
RMRSE Direct	MUTAG	0.12 ± 0.03	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01
	Cuneiform	0.24 ± 0.04	0.14 ± 0.05	0.09 ± 0.03	0.07 ± 0.01	0.07 ± 0.01	0.07 ± 0.01	0.07 ± 0.01
	SYNTHETIC	0.38 ± 0.01	0.29 ± 0.01	0.21 ± 0.00	0.17 ± 0.00	0.17 ± 0.00	0.17 ± 0.00	0.17 ± 0.00
	ENZYMES	0.15 ± 0.05	0.10 ± 0.04	0.09 ± 0.03	0.09 ± 0.03	0.09 ± 0.03	0.09 ± 0.03	0.09 ± 0.03
	IMDB-BINARY	0.29 ± 0.04	0.20 ± 0.04	0.12 ± 0.03	0.08 ± 0.02	0.06 ± 0.03	0.06 ± 0.03	0.06 ± 0.03
	CORA	0.44	0.34	0.23	0.12	0.09	0.08	0.08
RMRSE Neural	MUTAG	0.11 ± 0.01	0.10 ± 0.00	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01	0.10 ± 0.01
	Cuneiform	0.24 ± 0.04	0.14 ± 0.05	0.09 ± 0.03	0.07 ± 0.01	0.07 ± 0.01	0.07 ± 0.01	0.07 ± 0.01
	SYNTHETIC	0.37 ± 0.02	0.28 ± 0.02	0.21 ± 0.01	0.18 ± 0.00	0.18 ± 0.00	0.18 ± 0.00	0.18 ± 0.00
	ENZYMES	0.13 ± 0.04	0.10 ± 0.04	0.09 ± 0.03	0.09 ± 0.03	0.09 ± 0.03	0.09 ± 0.03	0.09 ± 0.04
	IMDB-BINARY	0.29 ± 0.04	0.20 ± 0.04	0.13 ± 0.03	0.08 ± 0.03	0.07 ± 0.03	0.07 ± 0.03	0.07 ± 0.03
	CORA	0.30	0.23	0.17	0.13	0.13	0.13	0.12

Table 1. Mean loss function results for $\kappa = 1.0$. Methods: RMSE=absolute loss, RMRSE=relative loss, Direct=embeddings optimized directly, Neural=embeddings regularized with a neural network

Method	Dataset	m=2	m=3	m=5	m=10	m=15	m=30	m=50
RMSE Direct	MUTAG	0.35 ± 0.19	0.21 ± 0.04	0.15 ± 0.03	0.04 ± 0.03	0.02 ± 0.01	0.02 ± 0.01	0.02 ± 0.00
	Cuneiform	0.50 ± 0.13	0.30 ± 0.11	0.16 ± 0.08	0.05 ± 0.02	0.03 ± 0.02	0.01 ± 0.00	0.01 ± 0.00
	SYNTHETIC	1.13 ± 0.04	0.87 ± 0.03	0.60 ± 0.01	0.39 ± 0.00	0.29 ± 0.00	0.24 ± 0.00	0.24 ± 0.00
	ENZYMES	0.39 ± 0.33	0.25 ± 0.18	0.17 ± 0.10	0.10 ± 0.07	0.08 ± 0.05	0.06 ± 0.04	0.06 ± 0.04
	IMDB-BINARY	0.23 ± 0.10	0.17 ± 0.07	0.12 ± 0.05	0.06 ± 0.04	0.03 ± 0.03	0.01 ± 0.02	0.02 ± 0.02
	CORA	3.25	2.56	1.76	0.93	0.69	0.48	0.39
RMSE Neural	MUTAG	0.30 ± 0.06	0.22 ± 0.04	0.20 ± 0.03	0.20 ± 0.04	0.20 ± 0.04	0.20 ± 0.04	0.20 ± 0.04
	Cuneiform	0.42 ± 0.08	0.29 ± 0.10	0.17 ± 0.07	0.09 ± 0.02	0.08 ± 0.01	0.08 ± 0.02	0.08 ± 0.02
	SYNTHETIC	0.79 ± 0.01	0.71 ± 0.01	0.58 ± 0.00	0.40 ± 0.01	0.37 ± 0.01	0.36 ± 0.01	0.36 ± 0.01
	ENZYMES	0.32 ± 0.17	0.26 ± 0.12	0.23 ± 0.10	0.22 ± 0.10	0.22 ± 0.09	0.22 ± 0.10	0.22 ± 0.10
	IMDB-BINARY	0.20 ± 0.07	0.17 ± 0.06	0.12 ± 0.05	0.08 ± 0.04	0.07 ± 0.04	0.06 ± 0.03	0.06 ± 0.03
	CORA	1.10	1.02	1.01	0.98	0.92	1.95	0.96
RMRSE Direct	MUTAG	0.10 ± 0.03	0.07 ± 0.01	0.05 ± 0.01	0.01 ± 0.01	0.00 ± 0.00	0.01 ± 0.00	0.01 ± 0.00
	Cuneiform	0.22 ± 0.03	0.14 ± 0.05	0.07 ± 0.04	0.02 ± 0.01	0.01 ± 0.01	0.00 ± 0.00	0.00 ± 0.00
	SYNTHETIC	0.36 ± 0.01	0.28 ± 0.01	0.21 ± 0.00	0.12 ± 0.00	0.09 ± 0.00	0.08 ± 0.00	0.08 ± 0.00
	ENZYMES	0.14 ± 0.05	0.09 ± 0.04	0.06 ± 0.03	0.04 ± 0.02	0.03 ± 0.02	0.03 ± 0.02	0.03 ± 0.02
	IMDB-BINARY	0.16 ± 0.06	0.14 ± 0.05	0.10 ± 0.04	0.04 ± 0.03	0.02 ± 0.02	0.01 ± 0.01	0.01 ± 0.02
	CORA	0.42	0.29	0.19	0.11	0.09	0.07	0.06
RMRSE Neural	MUTAG	0.09 ± 0.01	0.07 ± 0.01	0.06 ± 0.01	0.05 ± 0.01	0.05 ± 0.01	0.05 ± 0.01	0.05 ± 0.01
	Cuneiform	0.22 ± 0.03	0.14 ± 0.05	0.07 ± 0.03	0.03 ± 0.01	0.03 ± 0.01	0.03 ± 0.01	0.03 ± 0.01
	SYNTHETIC	0.32 ± 0.01	0.27 ± 0.01	0.21 ± 0.01	0.13 ± 0.01	0.11 ± 0.00	0.10 ± 0.00	0.10 ± 0.00
	ENZYMES	0.13 ± 0.04	0.09 ± 0.03	0.07 ± 0.02	0.06 ± 0.02	0.07 ± 0.02	0.07 ± 0.03	0.07 ± 0.03
	IMDB-BINARY	0.17 ± 0.05	0.14 ± 0.05	0.10 ± 0.04	0.05 ± 0.03	0.04 ± 0.02	0.04 ± 0.02	0.04 ± 0.02
	CORA	0.20	0.17	0.15	0.14	0.14	0.14	0.12

Table 2. Mean loss function results for κ optimized along with embeddings. Method: see Table 1

5.3 Graph drawing

Figure 2 illustrates vertex embeddings for different κ and loss types (absolute or relative). We selected the American Revolution [1] graph for visualization purposes, as the graph perfectly shows local node communities being represented in the embeddings' space. It is seen that minimization of the relative loss puts more emphasis on the local graph structure, while minimization of the absolute loss focuses more on its global structure. Also, with higher κ comes a tendency to compress local clusters of vertices.

5.4 Community detection in graphs

Methodology. Our study aimed to identify communities within graphs using a diverse array of unsupervised clustering algorithms applied to the embeddings of graph nodes generated through our methods. The clustering algorithms employed encompassed MeanShift [29], DBSCAN [16], HDBSCAN [33], Birch [59], OPTICS [3], AffinityPropagation [47] and AgglomerativeClustering [39]

We assessed the modularity score for all graphs, as this metric does not necessitate a graph with defined communities. For the Zachary Karate Club and American Football graphs, we additionally examined the ARS (adjusted rand

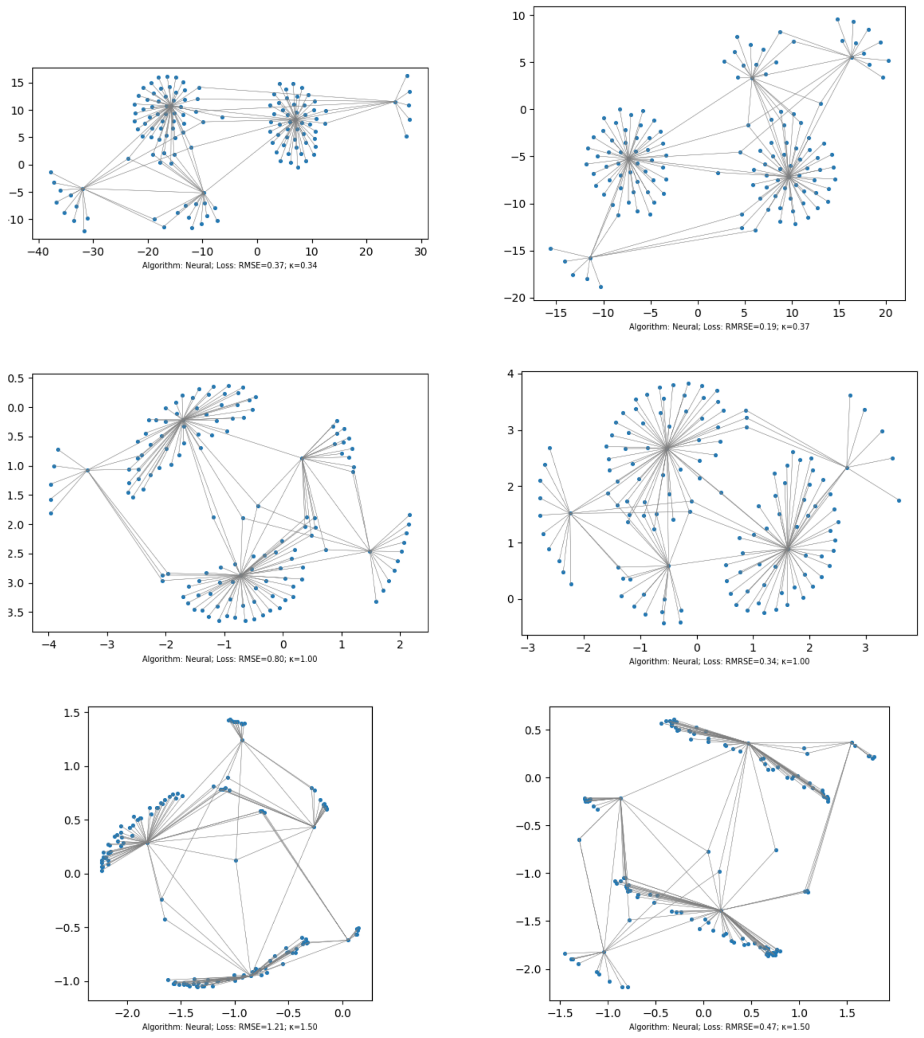

Fig. 2. American Revolution [1] graph visual representation ($m = 2$). *Left:* absolute error minimized. *Right:* relative error minimized. *Top:* $\kappa = auto \cong 0.4$, *Middle:* $\kappa = 1$, *Bottom:* $\kappa = 1.5$.

Method	Dataset	m=2	m=3	m=5	m=10	m=15	m=30	m=50
RMSE Direct	MUTAG	1.02 ± 0.06	1.13 ± 0.04	1.33 ± 0.09	1.81 ± 0.16	1.89 ± 0.07	1.88 ± 0.07	1.85 ± 0.08
	Cuneiform	0.62 ± 0.11	0.81 ± 0.14	1.06 ± 0.11	1.25 ± 0.02	1.34 ± 0.04	1.45 ± 0.11	1.46 ± 0.11
	SYNTHETIC	0.70 ± 0.01	0.72 ± 0.01	0.84 ± 0.01	1.41 ± 0.00	1.81 ± 0.01	1.95 ± 0.02	1.95 ± 0.02
	ENZYMES	1.02 ± 0.12	1.08 ± 0.07	1.19 ± 0.11	1.47 ± 0.22	1.66 ± 0.22	1.74 ± 0.19	1.70 ± 0.19
	IMDB-BINARY	0.39 ± 0.16	0.54 ± 0.24	0.75 ± 0.32	1.06 ± 0.45	1.32 ± 0.41	1.50 ± 0.29	1.48 ± 0.24
	CORA	1.24	1.22	1.20	1.18	1.16	1.24	1.27
RMSE Neural	MUTAG	1.03 ± 0.03	1.11 ± 0.04	1.16 ± 0.06	1.14 ± 0.06	1.16 ± 0.06	1.16 ± 0.06	1.17 ± 0.05
	Cuneiform	0.57 ± 0.16	0.81 ± 0.14	1.05 ± 0.11	1.18 ± 0.03	1.19 ± 0.04	1.20 ± 0.04	1.21 ± 0.04
	SYNTHETIC	0.38 ± 0.01	0.55 ± 0.01	0.84 ± 0.00	1.24 ± 0.01	1.31 ± 0.02	1.35 ± 0.01	1.35 ± 0.01
	ENZYMES	0.95 ± 0.10	1.01 ± 0.08	1.06 ± 0.10	1.07 ± 0.10	1.07 ± 0.10	1.07 ± 0.10	1.07 ± 0.10
	IMDB-BINARY	0.38 ± 0.16	0.54 ± 0.23	0.75 ± 0.29	0.99 ± 0.35	1.05 ± 0.33	1.09 ± 0.30	1.10 ± 0.29
	CORA	1.10	1.02	1.01	0.98	0.92	1.95	0.96
RMRSE Direct	MUTAG	1.10 ± 0.03	1.19 ± 0.03	1.40 ± 0.07	1.87 ± 0.12	1.94 ± 0.06	1.91 ± 0.05	1.84 ± 0.05
	Cuneiform	0.76 ± 0.08	0.97 ± 0.08	1.13 ± 0.06	1.26 ± 0.03	1.35 ± 0.05	1.45 ± 0.11	1.46 ± 0.11
	SYNTHETIC	0.69 ± 0.01	0.81 ± 0.01	1.11 ± 0.00	1.59 ± 0.00	1.90 ± 0.01	1.96 ± 0.00	1.91 ± 0.01
	ENZYMES	1.00 ± 0.08	1.09 ± 0.05	1.26 ± 0.11	1.63 ± 0.20	1.80 ± 0.18	1.83 ± 0.15	1.76 ± 0.13
	IMDB-BINARY	0.39 ± 0.16	0.56 ± 0.24	0.82 ± 0.35	1.15 ± 0.48	1.37 ± 0.43	1.52 ± 0.29	1.49 ± 0.25
	CORA	1.01	1.06	1.11	1.11	1.14	1.31	1.50
RMRSE Neural	MUTAG	1.11 ± 0.02	1.19 ± 0.03	1.31 ± 0.05	1.33 ± 0.04	1.33 ± 0.05	1.33 ± 0.04	1.35 ± 0.04
	Cuneiform	0.76 ± 0.09	0.97 ± 0.08	1.12 ± 0.06	1.22 ± 0.04	1.22 ± 0.04	1.23 ± 0.04	1.23 ± 0.04
	SYNTHETIC	0.58 ± 0.01	0.81 ± 0.02	1.11 ± 0.01	1.50 ± 0.01	1.58 ± 0.02	1.62 ± 0.01	1.61 ± 0.02
	ENZYMES	0.97 ± 0.09	1.08 ± 0.08	1.19 ± 0.12	1.22 ± 0.14	1.22 ± 0.13	1.22 ± 0.12	1.24 ± 0.13
	IMDB-BINARY	0.39 ± 0.15	0.57 ± 0.23	0.82 ± 0.32	1.12 ± 0.41	1.19 ± 0.38	1.22 ± 0.36	1.23 ± 0.36
	CORA	0.46	0.58	0.71	0.79	0.79	0.86	0.91

Table 3. Mean learned κ

index) and NMI (normalized mutual information) scores, which are valuable metrics for assessing how effectively an algorithm detects known communities.

We compared the results of our method with five widely used community detection algorithms: greedy modularity communities [14], Louvain communities [7], Kernighan Lin bisection [25], Girvan Newman [18], and asynchronous label propagation algorithm (asyn LPA)[42].

Results. Table 4 presents the Zachary Karate Club graph results. Our method outperforms other community detection algorithms in terms of ARS and NMI scores. Interestingly, the modularity score in this case does not align with the performance of the best method. Table 5 exhibits similar results but for the American Football graph. Our method also performs admirably, although slightly worse than the Girvan Newman method.

The modularity results for the TUDataset are detailed in Table 7. We compared these results with other algorithms, as shown in Table 6. Our method achieves competitive modularity scores, but not the highest ones. Notably, the method of designating the embedding does not impact the achieved modularity.

6 Conclusions

In this paper, we introduced a regularization method for graph vertex embeddings that preserves distances in the graph. This method uses a neural network to

Embed. method	Error	κ	m	Clustering algorithm	ARS	NMI	Modularity
-	-	-	-	Greedy Modularity Communities	0.57	0.56	0.38
-	-	-	-	Louvain communities	0.51	0.60	**0.42**
-	-	-	-	Kernighan Lin Bisection	0.77	0.68	0.37
-	-	-	-	Girvan Newman	0.77	0.73	0.36
-	-	-	-	Asyn Lpa Communities	0.66	0.65	0.38
Direct RMSE	0.45	0.72	2	MeanShift	0.88	0.84	0.37
Direct RMRSE	0.19	0.73	5	MeanShift	0.88	0.84	0.37
Direct RMSE	0.54	1.00	2	MeanShift	0.88	0.84	0.37
Direct RMRSE	0.26	1.00	2	AffinityPropagation	0.88	0.84	0.37
Direct RMSE	0.76	1.50	2	MeanShift	0.88	0.84	0.37
Direct RMRSE	0.37	1.50	2	MeanShift	0.88	0.84	0.37
Neural RMSE	0.37	0.81	3	MeanShift	0.88	0.84	0.37
Neural RMRSE	0.19	0.93	3	MeanShift	0.88	0.84	0.37
Neural RMSE	0.23	1.00	50	AffinityPropagation	**1.00**	**1.00**	0.36
Neural RMRSE	0.20	1.00	3	MeanShift	0.88	0.84	0.37
Neural RMSE	0.74	1.50	2	MeanShift	0.88	0.84	0.37
Neural RMRSE	0.08	1.50	50	AffinityPropagation	0.88	0.84	0.37

Table 4. Community detection in Zachary Karate Club

Embed. method	Error	κ	m	Clustering algorithm	ARS	NMI	Modularity
-	-	-	-	Greedy Modularity Communities	0.47	0.70	0.55
-	-	-	-	Louvain Communities	0.81	0.89	**0.60**
-	-	-	-	Kernighan Lin Bisection	0.14	0.38	0.35
-	-	-	-	Girvan Newman	**0.92**	**0.94**	0.36
-	-	-	-	Asyn Lpa Communities	0.75	0.87	0.58
Direct RMSE	0.46	0.54	10	AffinityPropagation	0.78	0.86	0.58
Direct RMRSE	0.24	0.51	5	HDBSCAN	0.89	0.92	0.58
Direct RMSE	0.36	1.00	15	AffinityPropagation	0.86	0.92	0.58
Direct RMRSE	0.18	1.00	15	AffinityPropagation	0.86	0.92	0.58
Direct RMSE	0.51	1.50	5	MeanShift	0.83	0.89	0.51
Direct RMRSE	0.13	1.50	50	AffinityPropagation	0.78	0.87	0.54
Neural RMSE	0.32	1.38	15	AffinityPropagation	0.83	0.90	0.57
Neural RMRSE	0.14	1.49	15	AffinityPropagation	0.80	0.88	0.54
Neural RMSE	0.36	1.00	15	AffinityPropagation	0.86	0.91	0.58
Neural RMRSE	0.18	1.00	15	AffinityPropagation	0.87	0.92	**0.60**
Neural RMSE	0.33	1.50	10	AffinityPropagation	0.81	0.90	0.55
Neural RMRSE	0.14	1.50	15	HDBSCAN	0.79	0.87	0.50

Table 5. Community detection in American Football

Dataset	GMC	LC	GN	KLB	ALC
MUTAG	**0.46 ± 0.06**	**0.46 ± 0.06**	**0.46 ± 0.06**	0.34 ± 0.05	0.41 ± 0.05
Cuneiform	**0.53 ± 0.03**	**0.53 ± 0.03**	0.47 ± 0.07	0.30 ± 0.07	0.47 ± 0.10
SYNTHETIC	**0.48 ± 0.00**	0.47 ± 0.01	0.44 ± 0.00	0.31 ± 0.01	0.35 ± 0.03
ENZYMES	0.57 ± 0.11	**0.58 ± 0.12**	**0.58 ± 0.13**	0.40 ± 0.08	0.54 ± 0.12
IMDB-BINARY	**0.30 ± 0.16**	**0.30 ± 0.16**	0.27 ± 0.16	0.20 ± 0.12	0.25 ± 0.17
CORA	0.81	**0.82**	0.81	0.41	0.5

Table 6. Mean modularity – graph community detection algorithms

	$\kappa = 1$				κ=auto			
Method	Dataset	Clustering algorithm	m	Modularity	Dataset	Clustering algorithm	m	Modularity
RMSE Direct	MUTAG	AffinityPropagation	50	0.43 ± 0.05	MUTAG	AffinityPropagation	5	0.44 ± 0.05
	Cuneiform	AffinityPropagation	2	0.44 ± 0.12	Cuneiform	AffinityPropagation	10	0.52 ± 0.07
	SYNTHETIC	AffinityPropagation	5	0.39 ± 0.02	SYNTHETIC	AffinityPropagation	5	0.39 ± 0.01
	ENZYMES	AffinityPropagation	10	0.54 ± 0.13	ENZYMES	AffinityPropagation	2	0.54 ± 0.13
	IMDB-BINARY	MeanShift	2	0.26 ± 0.16	IMDB-BINARY	HDBSCAN	3	0.25 ± 0.17
	CORA	AffinityPropagation	30	0.54	CORA	AffinityPropagation	50	0.57
RMSE Neural	MUTAG	AffinityPropagation	2	0.44 ± 0.05	MUTAG	AffinityPropagation	3	0.44 ± 0.05
	Cuneiform	AffinityPropagation	2	0.52 ± 0.07	Cuneiform	AffinityPropagation	50	0.52 ± 0.05
	SYNTHETIC	AffinityPropagation	5	0.40 ± 0.01	SYNTHETIC	AffinityPropagation	5	0.41 ± 0.00
	ENZYMES	AffinityPropagation	2	0.55 ± 0.12	ENZYMES	AffinityPropagation	2	0.55 ± 0.12
	IMDB-BINARY	MeanShift	2	0.26 ± 0.16	IMDB-BINARY	AffinityPropagation	2	0.26 ± 0.17
	CORA	AffinityPropagation	50	0.49	CORA	AffinityPropagation	15	0.58
RMRSE Direct	MUTAG	AffinityPropagation	5	0.44 ± 0.05	MUTAG	AffinityPropagation	3	0.44 ± 0.05
	Cuneiform	AffinityPropagation	10	0.53 ± 0.03	Cuneiform	AffinityPropagation	5	0.51 ± 0.09
	SYNTHETIC	AffinityPropagation	10	0.41 ± 0.00	SYNTHETIC	AffinityPropagation	5	0.40 ± 0.02
	ENZYMES	AffinityPropagation	5	0.55 ± 0.13	ENZYMES	AffinityPropagation	2	0.55 ± 0.12
	IMDB-BINARY	HDBSCAN	3	0.25 ± 0.17	IMDB-BINARY	AffinityPropagation	2	0.26 ± 0.18
	CORA	AffinityPropagation	30	0.64	CORA	AffinityPropagation	30	0.64
RMRSE Neural	MUTAG	AffinityPropagation	15	0.44 ± 0.05	MUTAG	AffinityPropagation	3	0.44 ± 0.05
	Cuneiform	AffinityPropagation	30	0.53 ± 0.04	Cuneiform	AffinityPropagation	5	0.52 ± 0.06
	SYNTHETIC	AffinityPropagation	50	0.41 ± 0.01	SYNTHETIC	AffinityPropagation	5	0.40 ± 0.01
	ENZYMES	AffinityPropagation	10	0.55 ± 0.13	ENZYMES	AffinityPropagation	2	0.55 ± 0.12
	IMDB-BINARY	AffinityPropagation	5	0.24 ± 0.18	IMDB-BINARY	AffinityPropagation	2	0.26 ± 0.18
	CORA	AffinityPropagation	15	0.58	CORA	AffinityPropagation	50	0.59

Table 7. Best mean modularity in different datasets

transform a column of the distance matrix into the embedding. In our experimental study, this regularization significantly improved the embeddings, especially when their dimension was low.

We also introduced a generalized measure of distance between the embeddings. With our proposed measure, the error of distance preservation by the embeddings was reduced by a large margin.

Finally, we performed a study on community detection in graphs, in which we compared results obtained by combining graph embeddings and clustering methods for numerical data with community detection algorithms dedicated to graphs. The analyzed combination achieved competitive results, although it yielded the best results only in the case of Zachary Karate Club.

References

1. American revolution network dataset – KONECT (Oct 2017), http://konect.cc/networks/brunson_revolution 6, 8, 9
2. Agrawal, A., Ali, A., Boyd, S.: Minimum-Distortion Embedding. Foundations and Trends® in Machine Learning **14**(3), 211–378 (2021) 2, 3, 4
3. Ankerst, M., Breunig, M., Kröger, P., Sander, J.: Optics: Ordering points to identify the clustering structure. Sigmod Record **28**, 49–60 (6 1999) 3, 8
4. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: NIPS. vol. 14. MIT Press (2001) 2
5. Berkhout, J.B., Poormoghadam, D., Yi, C., Kalsbeek, A., Meijer, O.C., Mahfouz, A.: An integrated single-cell rna-seq atlas of the mouse hypothalamic paraventricular nucleus links transcriptomic and functional types. Journal of Neuroendocrinology **36**, e13367 (2 2024) 3
6. Beygelzimer, A., Kakade, S., Langford, J.: Cover trees for nearest neighbor. In: ICML. pp. 97–104. ACM Press (2006) 1
7. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment **2008**(10), P10008 (2008) 3, 10
8. Böhm, J.N., Berens, P., Kobak, D.: Attraction-repulsion spectrum in neighbor embeddings. The Journal of Machine Learning Research **23**(1), 4118–4149 (2022) 1
9. Borgwardt, K., Ghisu, E., Llinares-López, F., O'Bray, L., Rieck, B.: Graph Kernels: State-of-the-Art and Future Challenges. Foundations and Trends® in Machine Learning **13**(5-6), 531–712 (2020) 2
10. Bourgain, J.: On lipschitz embedding of finite metric spaces in hilbert space. Israel Journal of Mathematics **52**, 46–52 (1985) 2
11. Chamberlain, B.P., Clough, J., Deisenroth, M.P.: Neural Embeddings of Graphs in Hyperbolic Space. In: ICLR (2017) 3
12. Chari, T., Pachter, L.: The specious art of single-cell genomics. PLOS Computational Biology **19**, e1011288 (8 2023) 3
13. Chen, Z., Li, L., Bruna, J.: Supervised Community Detection with Line Graph Neural Networks. In: ICLR (2018) 3
14. Clauset, A., Newman, M.E., Moore, C.: Finding community structure in very large networks. Physical review E **70**(6), 066111 (2004) 10
15. Dwivedi, V.P., Joshi, C.K., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Benchmarking Graph Neural Networks. Journal of Machine Learning Research **24**(43), 1–48 (2023) 3
16. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: International Conference on Knowledge Discovery and Data Mining. pp. 226–231 (1996) 3, 8
17. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: ICML. pp. 1263–1272 (2017) 3
18. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences **99**(12), 7821–7826 (2002) 3, 6, 10
19. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: International Conf. on Knowledge Discovery & Data Mining. pp. 855–864 (2016) 3
20. Hoffmann, M., Henninger, J., Veith, J., Richter, L., Judkewitz, B.: Blazed oblique plane microscopy reveals scale-invariant inference of brain-wide population activity. Nature Communications 2023 14:1 **14**, 1–11 (12 2023) 3

21. Indyk, P.: Algorithmic applications of low-distortion geometric embeddings. In: IEEE Symposium on Foundations of Computer Science. pp. 10–33 (2001) 2
22. Ji, P., Zhang, T., Li, H., Salzmann, M., Reid, I.: Deep subspace clustering networks. In: NIPS. vol. 30 (2017) 3
23. Johnson, W., Lindenstrauss, J.: Extensions of lipschitz maps into a hilbert space. Contemporary Mathematics **26**, 189–206 (1 1984) 2
24. Jumper, J., and others: Highly accurate protein structure prediction with AlphaFold. Nature **596**(7873), 583–589 (2021) 2
25. Kernighan, B.W., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell System Technical Journal **49**(2), 291–307 (1970) 10
26. Kipf, T.N., Welling, M.: Variational Graph Auto-Encoders (2016), arXiv:1611.07308 1
27. Kipf, T.N., Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks (2017), arXiv:1609.02907 3
28. Klimovskaia, A., Lopez-Paz, D., Bottou, L., Nickel, M.: Poincaré maps for analyzing complex hierarchies in single-cell data. Nature Communications **11**(1), 2966 (2020) 3
29. Koohpayegani, S.A., Tejankar, A., Pirsiavash, H.: Mean shift for self-supervised learning. In: ICCV. pp. 10326–10335 (10 2021) 3, 8
30. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: ICML. vol. 4 (5 2014) 3
31. Linial, N., London, E., Rabinovich, Y.: The geometry of graphs and some of its algorithmic applications. Combinatorica **15**(2), 215–245 (1995) 2, 3
32. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of Machine Learning Research **9**, 2579–2605 (2008) 2
33. Malzer, C., Baum, M.: A hybrid approach to hierarchical density-based cluster selection. In: IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (9 2020) 3, 8
34. McInnes, L., Healy, J., Melville, J.: Umap: Uniform manifold approximation and projection for dimension reduction (2020) 2
35. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) NIPS (2013) 3
36. Morris, C., Dym, N., Maron, H., Ceylan, I.I., Frasca, F., Levie, R., Lim, D., Bronstein, M., Grohe, M., Jegelka, S.: Future Directions in Foundations of Graph Machine Learning (2024), arXiv:2402.02287 3
37. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: Tudataset: A collection of benchmark datasets for learning with graphs. In: ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020) (2020), www.graphlearning.io 6
38. Morselli Gysi, D., do Valle, I., Zitnik, M., Ameli, A., Gan, X., Varol, O., Ghiassian, S.D., Patten, J.J., Davey, R.A., Loscalzo, J., Barabási, A.L.: Network medicine framework for identifying drug-repurposing opportunities for COVID-19. Proceedings of the National Academy of Sciences **118**(19), e2025581118 (2021) 2
39. Müllner, D.: Modern hierarchical, agglomerative clustering algorithms (2011), https://arxiv.org/abs/1109.2378 3, 8
40. Nickel, M., Kiela, D.: Poincaré Embeddings for Learning Hierarchical Representations. In: Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017) 3

41. Pothen, A., Simon, H.D., Liou, K.P.: Partitioning Sparse Matrices with Eigenvectors of Graphs. SIAM Journal on Matrix Analysis and Applications **11**(3), 430–452 (1990) 3
42. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Physical Review E **76**(3) (Sep 2007) 10
43. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**, 2323–2326 (12 2000) 2
44. Sala, F., De Sa, C., Gu, A., Ré, C.: Representation tradeoffs for hyperbolic embeddings. In: ICML. pp. 4460–4469. PMLR (2018) 3
45. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data articles. AI Magazine **29**, 93–106 (09 2008) 6
46. Su, X., Xue, S., Liu, F., Wu, J., Yang, J., Zhou, C., Hu, W., Paris, C., Nepal, S., Jin, D., Sheng, Q.Z., Yu, P.S.: A Comprehensive Survey on Community Detection with Deep Learning. IEEE Transactions on Neural Networks and Learning Systems pp. 1–21 (2024) 3
47. Sun, L., Liu, R., Xu, J., Zhang, S., Tian, Y.: An affinity propagation clustering method using hybrid kernel function with lle. IEEE Access **PP**, 1 (11 2018) 3, 8
48. Tang, J., Liu, J., Zhang, M., Mei, Q.: Visualization large-scale and high-dimensional data (2016), http://arxiv.org/abs/1602.00370 2
49. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2323 (12 2000) 2, 3
50. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, t.L., Polosukhin, I.: Attention is all you need. In: NIPS. vol. 30 (2017) 1
51. Velicković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph Attention Networks. In: ICLR (2018) 3
52. Velicković, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D.: Deep Graph Infomax (2018), arXvi:1809.10341 3
53. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A., Bottou, L.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of machine learning research **11**(12) (2010) 1
54. Wang, X., Cui, P., Wang, J., Pei, J., Zhu, W., Yang, S.: Community preserving network embedding. In: AAAI Conference on Artificial Intelligence. vol. 31 (2017) 3
55. Weinberger, K., Kilian, Saul, Lawrence: Unsupervised learning of image manifolds by semidefinite programming. International Journal of Computer Vision **70**, 77 (10 2006) 2
56. Wu, L., Cui, P., Pei, J., Zhao, L. (eds.): Graph Neural Networks: Foundations, Frontiers, and Applications. Springer Nature Singapore (2022) 2
57. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. In: ACM SIGKDD Workshop on Mining Data Semantics. pp. 1–8. ACM (2012) 3
58. Zachary, W.: An information flow model for conflict and fission in small groups. J. of Anthropol. Res. **33**, 452–473 (1977) 6
59. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: an efficient data clustering method for very large databases. SIGMOD Rec. **25**, 103–114 (6 1996) 3, 8
60. Zhang, Y.J., Yang, K.C., Radicchi, F.: Systematic comparison of graph embedding methods in practical tasks. Physical Review E **104**(4), 044315 (2021) 2