applied sciences

MDPI

Article

Community Detection in Attributed Networks via Adaptive Graph Propagation and Density Peaks

Teng Zhang * and Xupeng Ren

School of Computer Science and Artificial Intelligence, Lanzhou University of Technology, Lanzhou 730050, China; renxp@lut.edu.cn
* Correspondence: ztbaozi@126.com

Abstract

Graph propagation can effectively fuse network topology and node attributes to conduct community detection in attributed networks. However, selecting a suitable propagation depth remains challenging because insufficient propagation limits information aggregation, whereas excessive propagation may reduce representation discriminability. This study proposes a training-free community detection framework that combines adaptive propagation depth selection, density-peak initialization, and centroid-based refinement. The propagation depth is selected by contrasting representation distances between fixed sets of informative node pairs constructed from the original graph and attributes, without ground-truth labels or intermediate clustering. The resulting representation is used for density-peak initialization, followed by centroid-based refinement with an optional community-size penalty. Experiments on five benchmark attributed networks under the prescribed-community-count setting demonstrate competitive clustering performance against established baselines.

Keywords: attributed networks; community detection; propagation-depth selection; density peak clustering

1. Introduction

Community detection aims to identify meaningful groups of nodes in networked systems [1,2]. Attributed networks, in which nodes are associated with descriptive features, arise in diverse domains, including social, biological, and collaboration networks [3,4]. Network topology and node attributes can provide complementary information, making their effective integration an important research direction in community detection.

In recent years, graph convolution-based methods have demonstrated their effectiveness in integrating network structure and node attributes [5]. However, these methods often involve complex training procedures, and their propagation depth directly determines the range of information aggregation [6,7]. Shallow propagation may fail to fully exploit structural information, whereas excessively deep propagation can blur distinctions between node representations across community boundaries, leading to oversmoothing [8,9]. Meanwhile, node labels in real-world networks are often scarce or entirely unavailable. Therefore, adaptively selecting an appropriate propagation depth without relying on ground-truth node labels has become a key challenge in community detection in attributed networks.

Several studies provide relevant foundations for adaptive propagation and community detection. AGC selects the propagation order using intra-cluster distances obtained from intermediate clustering results [6]. MADGap characterizes oversmoothing through the

Check for updates

Received: 11 October 2026
Revised: 15 November 2026
Accepted: 26 November 2026
Published: 28 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

distance contrast between remote and neighboring node representations, while MADReg incorporates this contrast into a GNN training objective [9]. GATFELPA combines adaptive aggregation, similarity preservation, and density-peak-based community detection [10]. Building on these studies, we investigate how topology- and attribute-informed node pairs can guide propagation-depth selection along a fixed linear propagation sequence without requiring intermediate clustering.

Specifically, we construct two sets of informative node pairs from the original graph and attributes: adjacent pairs with low attribute similarity and nonadjacent pairs that are structurally distant and have low attribute similarity. Both sets remain fixed throughout the propagation-depth scan, providing a consistent basis for evaluating candidate depths. The representation at the selected depth is subsequently reduced in dimension and used for density-peak initialization and centroid-based community refinement. The framework requires no neural-network parameter training, although it involves numerical decomposition, algorithmic hyperparameters, and iterative refinement. It takes the number of communities as a prescribed input rather than estimating it automatically. The main contributions are summarized as follows:

1. We develop an adaptive propagation-depth selection procedure based on fixed node pairs selected jointly using network topology and original attribute similarities. A pair-averaged remote–neighbor distance contrast evaluates candidate depths without requiring ground-truth node labels or intermediate clustering.
2. We integrate the depth-selection procedure with density-peak initialization and centroid-based refinement into a training-free framework for attributed-network community detection. The refinement combines standardized cosine-similarity scores with an optional community-size penalty.
3. We evaluate the framework on five benchmark attributed networks through comparisons with representative baselines, propagation-depth analysis, and component ablations. The results characterize its comparative performance under a protocol restricted to the largest connected component and using the reference number of communities.

2. Related Work

2.1. Topology-Based Community Detection

Topology-based community detection identifies communities from network connectivity. A widely used quality function is modularity, which measures the difference between the observed fraction of within-community edges and its expectation under a null model [11]. Representative modularity optimization methods include spectral approaches [12], greedy agglomeration [13], and the Louvain method [14]. Louvain alternates between local node moves that increase modularity and aggregation of the resulting communities, making it practical for analyzing large networks.

Other approaches identify network groups through local label updates, information flow, or probabilistic modeling. Label propagation assigns each node a label occurring most frequently among its neighbors and is computationally efficient, although its results can depend on update order and random tie-breaking [15]. Infomap identifies communities by minimizing the description length of random walks [16], whereas stochastic block models characterize connectivity through group-dependent edge probabilities [17].

In their topology-only formulations, the methods reviewed above infer communities from network connectivity without directly incorporating node attributes. Louvain and the LPA also do not require the number of communities to be specified in advance [14,15]. In contrast, the proposed framework combines network topology with node attributes and takes a prescribed community count as input.

2.2. Community Detection in Attributed Networks

Established approaches to attributed-network community detection include graph augmentation, probabilistic modeling, and matrix factorization. Graph-augmentation approaches encode attribute information through additional vertices or edges. SA-Cluster constructs an attribute-augmented graph and combines structural and attribute similarities using random-walk-based distances with adaptively adjusted weights [18]. Its incremental extension reduces the cost of updating these distances during weight adjustment [19]. CODICIL combines content-based and structural connections to construct a graph for subsequent clustering [20].

Probabilistic approaches explicitly model the relationship between community memberships, links, and attributes. CESNA jointly models network edges and binary node attributes through overlapping community memberships [4]. BAGC uses a Bayesian model to infer communities from structural and attribute information [21]. These methods make the relationship between the two information sources explicit, while their suitability depends on the compatibility of their modeling assumptions with the observed data.

Matrix-factorization methods provide another approach to integrating network topology and node attributes. SCI uses non-negative matrix factorization to jointly estimate community memberships and community attributes, supporting both community detection and semantic interpretation [22]. Jin et al. [2] review broader methodological developments, while Chunaev [3] surveys attributed-network community detection according to how and when structural and attribute information is combined. Our framework integrates these sources through a fixed linear propagation operator, using informative node pairs to select the propagation depth before density-peak initialization and centroid-based refinement.

2.3. Graph Representation Learning and Adaptive Propagation

Network representation learning maps nodes into a low-dimensional space for downstream tasks, including clustering. DeepWalk and node2vec learn node embeddings from random-walk sequences [23,24], whereas LINE preserves first-order and second-order proximity through dedicated objectives [25]. Their original objectives capture structural relationships rather than directly optimize community assignments.

Graph neural networks integrate node attributes with topology through feature transformation and neighborhood aggregation. GCN uses normalized graph propagation [5], while GAT learns attention weights for neighboring nodes [26]. For unsupervised representation learning, graph autoencoders and variational graph autoencoders reconstruct connectivity from latent embeddings [27], and ARGA adds adversarial regularization [28]. DAEGC and SDCN further couple representation learning with clustering-oriented objectives [7,29]. These models typically require iterative parameter optimization and choices of architecture and training objectives.

Propagation depth affects the balance between information aggregation and representation discriminability. Repeated aggregation extends the receptive field, but excessive smoothing can make node representations less distinguishable. Li et al. interpreted graph convolution through Laplacian smoothing [8], and Oono and Suzuki analyzed the loss of expressive power in deep graph neural networks under specified conditions [30]. Approaches to mitigating this effect include initial residual connections and identity mapping in GCNII [31], edge dropping in DropEdge [32], and representation normalization in PairNorm [33]. GPR-GNN learns weights for different propagation orders under a task-specific training objective [34].

For unsupervised attributed graph clustering, AGC selects propagation order from intermediate clustering results [6], while GATFELPA combines adaptive aggregation, graph

attention, and density-peak-based enhanced label propagation [10]. MADGap characterizes oversmoothing through a remote–neighbor distance contrast, which also underlies MADReg regularization [9]. Building on this contrast, our method selects propagation depth using pair-averaged distances over fixed node pairs filtered jointly by topology and original attribute similarity. Its distinction from MADGap lies in the pair construction and the criterion's role: selecting a depth along a fixed linear propagation sequence before clustering, rather than diagnosing oversmoothing or regularizing GNN training. This selection requires neither intermediate partitions nor neural-network parameter optimization.

2.4. Density-Peak-Based Clustering and Community Detection

Density peak clustering identifies candidate centers through high local density and large separation from higher-density points [35]. Extensions to community detection adapt density and distance measures to graph structure or node representations [36,37].

Recent studies have incorporated density peaks into multistage community detection frameworks. Tang et al. combine density-peak-based enhanced label propagation with graph attention and adaptive aggregation in GATFELPA [10]. Jia et al. propose DPCNCM, which estimates the community count and selects initial centers in a joint structure–attribute embedding before refining overlapping memberships through neutrosophic C-means [38]. Gao et al. (2026) develop D-means, combining topology-based density-peak center selection with K-means community detection and a center-count threshold derived from Chebyshev's inequality [39].

In our framework, density peaks serve as an initialization mechanism in the compact representation obtained after adaptive propagation-depth selection. A prescribed number of centers is selected, and non-center nodes are assigned to their nearest selected center before centroid-based refinement. This initial assignment differs from the higher-density-neighbor label inheritance used in the original DPC algorithm. The framework targets non-overlapping communities with a given count, focusing on the integration of propagation-depth selection, representation-based initialization, and subsequent refinement.

3. The Proposed Method

This section first formulates the community detection problem in attributed networks and introduces the notation used throughout the paper. It then presents the proposed framework, which comprises three stages: graph propagation with adaptive depth selection, density-peak initialization, and centroid-based refinement, as illustrated in Figure 1.

3.1. Problem Formulation and Notation

Let $G = (V, E, \mathbf{X})$ be an undirected, unweighted attributed graph, where $V = \{v_1, \ldots, v_N\}$ is the node set and E is the edge set. The adjacency matrix is $\mathbf{A} = [a_{ij}] \in \{0,1\}^{N\times N}$, where $a_{ij} = 1$ if $(v_i, v_j) \in E$, and 0 otherwise. The degree matrix is $\mathbf{D} = \mathrm{diag}(d_1, \ldots, d_N)$, with $d_i = \sum_{j=1}^{N} a_{ij}$. The node attribute matrix is $\mathbf{X} = [\mathbf{x}_1^\top; \ldots; \mathbf{x}_N^\top] \in \mathbb{R}^{N\times F}$, where $\mathbf{x}_i \in \mathbb{R}^F$ is the attribute vector of node v_i and F is the number of attributes.

Given a prescribed community count $C \in \{1, \ldots, N\}$, the task is to obtain a community assignment $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_C\}$ using both network topology and node attributes, subject to

$$\bigcup_{c=1}^{C} \mathcal{C}_c = V, \qquad \mathcal{C}_c \cap \mathcal{C}_{c'} = \varnothing \quad (c \neq c'), \qquad C_{\mathrm{out}} = |\{c \in \{1, \ldots, C\} : \mathcal{C}_c \neq \varnothing\}| \leq C. \quad (1)$$

The nonempty sets among $\{\mathcal{C}_1, \ldots, \mathcal{C}_C\}$ form the final partition, and C_{out} denotes their number. Empty $\mathcal{C}_c$ may arise during synchronous refinement because no explicit empty-community repair is performed. Under the node ordering $V = \{v_1, \ldots, v_N\}$, $\mathbf{y} = (y_1, \ldots, y_N)^\top$ is the community-assignment vector, where $y_i = c$ if and only if $v_i \in \mathcal{C}_c$.

The count C is supplied rather than estimated, and ground-truth labels are used only for post hoc evaluation.

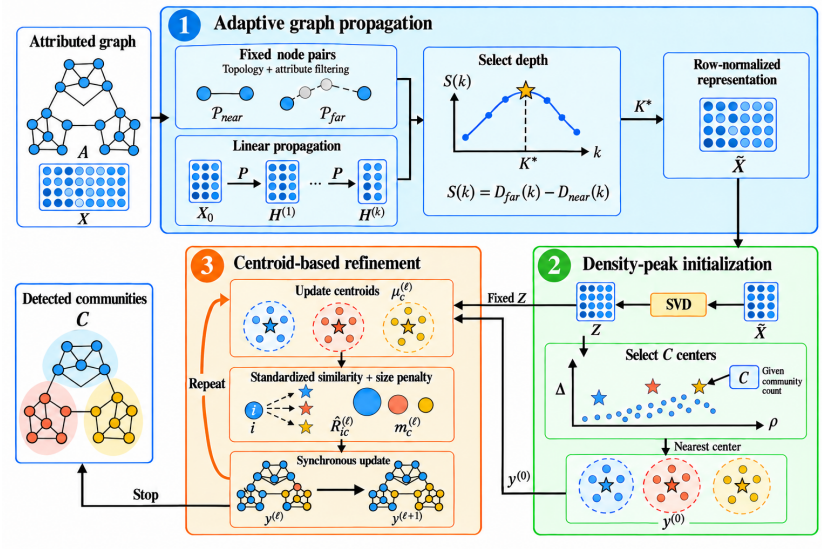

Figure 1. Overview of the proposed framework: adaptive graph propagation, density-peak initialization, and centroid-based refinement. Stars mark the selected propagation depth in stage 1, initial community centers in stage 2, and community centroids in stage 3. Node colors in the community diagrams distinguish communities, while background colors distinguish the three stages.

3.2. Graph Propagation and Adaptive Depth Selection

3.2.1. Linear Graph Propagation

Node attributes and network topology are integrated through linear propagation with a fixed, symmetrically normalized operator [5]:

$$\mathbf{H}^{(k+1)} = \mathbf{P}\mathbf{H}^{(k)}, \qquad \mathbf{P} = \widetilde{\mathbf{D}}^{-1/2}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-1/2}, \tag{2}$$

where $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ includes self-loops and $\widetilde{\mathbf{D}}$ is its diagonal degree matrix, with $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$. Here, $\mathbf{H}^{(k)} \in \mathbb{R}^{N\times F}$ denotes the representation matrix after k steps. The operator contains no trainable parameters. Self-loops include each node's current representation in the aggregation. Symmetric normalization assigns $P_{ij} = \widetilde{A}_{ij}/\sqrt{\widetilde{D}_{ii}\widetilde{D}_{jj}}$, accounting for both endpoint degrees and yielding a symmetric propagation operator. By comparison, random-walk normalization $\widetilde{\mathbf{D}}^{-1}\widetilde{\mathbf{A}}$ has unit row sums but is generally asymmetric.

The initial state is $\mathbf{H}^{(0)} = \mathbf{X}_0$, where each nonzero row of $\mathbf{X}$ is normalized to unit ℓ_2 norm and zero rows remain unchanged. This gives nonzero input vectors equal weight in terms of their norms before aggregation and emphasizes attribute directions, consistent with the subsequent cosine-based comparisons. The tradeoff is the loss of potentially informative input-magnitude differences. After k steps,

$$\mathbf{H}^{(k)} = \mathbf{P}^k\mathbf{X}_0, \tag{3}$$

which incorporates attribute information from nodes up to k hops away. A separate row-normalized copy $\widehat{\mathbf{H}}^{(k)}$ is used for depth evaluation; it is not fed back into the propagation recurrence.

Larger depths extend the aggregation range but may reduce representation discriminability through oversmoothing [8]. The following subsection introduces a criterion for selecting the depth without ground-truth node labels or intermediate clustering.

3.2.2. Adaptive Propagation Depth Selection

Propagation depth is selected using two sets of node pairs constructed from the original topology and normalized attributes. Each undirected pair (v_i, v_j) is considered once, with $i < j$. Attribute similarity is measured by $s_{ij} = \mathbf{x}_{0,i}^{\top}\mathbf{x}_{0,j}$, where $\mathbf{x}_{0,i}$ is the normalized attribute vector of node v_i. The same two sets of node pairs are used to evaluate all candidate propagation depths.

Hard-neighbor pairs ($\mathcal{P}_{\text{near}}$). Adjacent pairs with low original attribute similarity are selected to probe the smoothing of local attribute differences. The pairs are ranked by increasing s_{ij}, and a prescribed fraction of the lowest-similarity pairs is retained, subject to a maximum pair count, to form $\mathcal{P}_{\text{near}}$. The threshold θ_{near} is defined as the largest similarity among the retained pairs:

$$\theta_{\text{near}} = \max_{(v_i,v_j)\in\mathcal{P}_{\text{near}}} s_{ij}. \tag{4}$$

Structurally remote pairs ($\mathcal{P}_{\text{far}}$). Nonadjacent pairs with large shortest-path distances and low original attribute similarity are selected to probe representation separation. Candidate pairs are sampled from nonadjacent nodes, and the distance threshold τ_{far} is determined from their shortest-path-distance distribution, where $d_G(i,j)$ denotes the shortest-path distance. Among these sampled candidates satisfying $d_G(i,j) \geq \tau_{\text{far}}$, a prescribed fraction with the lowest attribute similarities is retained, subject to a maximum pair count, to form $\mathcal{P}_{\text{far}}$. The threshold θ_{far} is then defined as

$$\theta_{\text{far}} = \max_{(v_i,v_j)\in\mathcal{P}_{\text{far}}} s_{ij}. \tag{5}$$

The same candidate sample is used to determine τ_{far} and select $\mathcal{P}_{\text{far}}$.

Similarity ties are resolved by a fixed ordering of node pairs. Both retained sets must be nonempty. For each candidate depth k, the propagated representation $\mathbf{H}^{(k)}$ is obtained from Equation (3). A row-normalized copy is used for distance evaluation:

$$\widehat{\mathbf{H}}_{i,:}^{(k)} = \begin{cases} \dfrac{\mathbf{H}_{i,:}^{(k)}}{\|\mathbf{H}_{i,:}^{(k)}\|_2}, & \|\mathbf{H}_{i,:}^{(k)}\|_2 > 0, \\ \mathbf{0}^{\top}, & \text{otherwise.} \end{cases} \tag{6}$$

Let $\widehat{\mathbf{h}}_i^{(k)}$ denote the column vector corresponding to row i of $\widehat{\mathbf{H}}^{(k)}$. The pairwise dissimilarity is

$$\delta_{ij}^{(k)} = 1 - \widehat{\mathbf{h}}_i^{(k)\top}\widehat{\mathbf{h}}_j^{(k)}. \tag{7}$$

This equals the cosine distance when both representations are nonzero. The mean distances over the two fixed pair sets are

$$D_{\text{near}}(k) = \frac{1}{|\mathcal{P}_{\text{near}}|}\sum_{(v_i,v_j)\in\mathcal{P}_{\text{near}}} \delta_{ij}^{(k)}, \qquad D_{\text{far}}(k) = \frac{1}{|\mathcal{P}_{\text{far}}|}\sum_{(v_i,v_j)\in\mathcal{P}_{\text{far}}} \delta_{ij}^{(k)}. \tag{8}$$

Each retained pair contributes equally to its respective mean, including pairs with zero dissimilarity.

The depth-selection score is

$$S(k) = D_{\text{far}}(k) - D_{\text{near}}(k). \tag{9}$$

A higher score indicates a larger mean distance for remote pairs relative to adjacent pairs. The selected propagation depth is

$$K^* = \min\left(\arg\max_{k\in\{0,\ldots,K_{\max}\}} S(k)\right). \tag{10}$$

All candidate depths are evaluated, and the smallest depth is chosen when multiple candidates attain the maximum score. Depth selection requires neither ground-truth node labels nor intermediate clustering. The complete procedure is summarized in Algorithm A1.

The selected pairs do not necessarily correspond to within-community and between-community relationships: adjacent nodes may belong to different communities, and remote nodes may belong to the same community. Moreover, the difference of means does not capture the dispersion or overlap of the distance distributions. The score therefore serves as a heuristic for depth selection and does not guarantee optimal external clustering scores.

3.3. Density-Peak-Based Initialization

Given the selected propagation depth K^*, the final propagated representation is

$$\widetilde{\mathbf{X}} = \widehat{\mathbf{H}}^{(K^*)}, \tag{11}$$

where row normalization follows Equation (6). To reduce the dimensionality used in subsequent distance calculations, truncated singular value decomposition (SVD) is applied:

$$\widetilde{\mathbf{X}} \approx \mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^\top, \tag{12}$$

where r is the retained dimension. The compact representation is

$$\mathbf{Z} = \mathbf{U}_r\mathbf{\Sigma}_r \in \mathbb{R}^{N\times r}. \tag{13}$$

Each nonzero row is subsequently normalized to unit ℓ_2 norm, while zero rows remain unchanged. Let $\mathbf{z}_i$ denote the resulting vector of node v_i. The pairwise dissimilarity is

$$\delta_{ij} = 1 - \mathbf{z}_i^\top\mathbf{z}_j, \qquad i \neq j, \tag{14}$$

with $\delta_{ii} = 0$. For two nonzero representations, this equals the cosine distance.

Following the density peak clustering (DPC) principle, candidate centers are identified by high local density and large separation from higher-density nodes [35]. Here, local density is estimated from distances to nearby nodes in the compact representation space. Let $\mathcal{N}_q^{\mathbf{Z}}(i)$ denote the q nearest nodes other than v_i, with $1 \le q \le N-1$. For a positive mean neighbor distance, the local density is

$$\rho_i = \left(\frac{1}{q}\sum_{j\in\mathcal{N}_q^{\mathbf{Z}}(i)}\delta_{ij}\right)^{-1}. \tag{15}$$

A larger ρ_i indicates a more compact local neighborhood.

For a node below the maximum density, its relative distance is the distance to the nearest strictly higher-density node:

$$\Delta_i = \min_{j:\rho_j>\rho_i}\delta_{ij}, \qquad \rho_i < \max_l \rho_l. \tag{16}$$

For every node attaining the maximum density, the relative distance is defined as

$$\Delta_i = \max_{j \neq i} \delta_{ij}, \qquad \rho_i = \max_l \rho_l. \tag{17}$$

When $\max_j \Delta_j > 0$, the density-peak score is

$$\gamma_i = \frac{\rho_i}{\max_j \rho_j} \frac{\Delta_i}{\max_j \Delta_j}. \tag{18}$$

Given the prescribed community count C, the highest-density node is first selected as a center, with ties resolved by a fixed node order. The remaining $C-1$ centers are selected from the other nodes in decreasing order of γ_i, using the same order to resolve score ties. Let the selected center indices be

$$\mathcal{O} = \{o_1, \ldots, o_C\}. \tag{19}$$

Each center initializes its own community, $y_{o_c}^{(0)} = c$, and each non-center node is assigned to its nearest selected center:

$$y_i^{(0)} = \arg\min_{1 \leq c \leq C} \delta_{i,o_c}, \qquad i \notin \mathcal{O}. \tag{20}$$

Distance ties are resolved by center order. Each initial community therefore contains its selected center.

3.4. Centroid-Based Community Refinement

Starting from density-peak initialization, community assignments are refined by alternating centroid updates and synchronous node reassignment. The compact representations $\mathbf{z}_i$ remain fixed throughout refinement and already incorporate both network topology and node attributes.

Let $\mathbf{y}^{(\ell)}$ denote the assignment vector at iteration ℓ. The membership indicators and community sizes are

$$Y_{ic}^{(\ell)} = \begin{cases} 1, & y_i^{(\ell)} = c, \\ 0, & \text{otherwise}, \end{cases} \qquad m_c^{(\ell)} = \sum_{i=1}^{N} Y_{ic}^{(\ell)}. \tag{21}$$

The centroid of community $\mathcal{C}_c$ is

$$\boldsymbol{\mu}_c^{(\ell)} = \frac{\sum_{i=1}^{N} Y_{ic}^{(\ell)} \mathbf{z}_i}{\max\{m_c^{(\ell)}, 1\}}. \tag{22}$$

Each nonzero centroid is normalized to unit ℓ_2 norm, yielding $\widehat{\boldsymbol{\mu}}_c^{(\ell)}$; zero centroids remain unchanged. The node–centroid similarity is

$$R_{ic}^{(\ell)} = \mathbf{z}_i^{\top} \widehat{\boldsymbol{\mu}}_c^{(\ell)}, \tag{23}$$

which equals cosine similarity when both vectors are nonzero.

Before combining similarity with the community-size penalty, each node's similarities are standardized across the C candidate communities:

$$\widehat{R}_{ic}^{(\ell)} = \frac{R_{ic}^{(\ell)} - \bar{R}_i^{(\ell)}}{\max\{\sigma_{R_i}^{(\ell)}, \varepsilon_R\}}, \tag{24}$$

where

$$\bar{R}_i^{(\ell)} = \frac{1}{C}\sum_{c=1}^{C} R_{ic}^{(\ell)}, \qquad \sigma_{R_i}^{(\ell)} = \sqrt{\frac{1}{C}\sum_{c=1}^{C}\left(R_{ic}^{(\ell)} - \bar{R}_i^{(\ell)}\right)^2}.$$

Here, $\varepsilon_R > 0$ prevents division by zero. If all candidate similarities are identical, their standardized values are zero. Standardization adjusts the scale of the similarity term while preserving each node's ranking of candidate communities.

The assignment score is

$$S_{ic}^{(\ell)} = \alpha \widehat{R}_{ic}^{(\ell)} - \beta \log\left(m_c^{(\ell)} + 1\right), \tag{25}$$

where $\alpha, \beta \geq 0$ control the contributions of similarity and the size penalty, respectively. The penalty lowers the scores of currently larger communities, but does not enforce equal community sizes. It may discourage excessive concentration in a few communities, although it can also penalize genuinely large communities. In the weight-sensitivity experiment, $\beta = 1 - \alpha$, so the endpoint $\alpha = 1, \beta = 0$ removes the community-size penalty.

For $\alpha > 0$ and $\beta = 0$, standardization leaves the assignment maximizers unchanged. The normalized-centroid updates and cosine-based assignments therefore follow the spherical K-means structure [40]. Their equivalence requires matched initialization and consistent handling of ties, zero centroids, empty communities, and stopping conditions. When $\beta > 0$, standardization affects the relative influence of the size penalty.

All scores are computed from the partition at iteration ℓ, after which node labels are updated synchronously:

$$y_i^{(\ell+1)} = \underset{1 \leq c \leq C}{\arg\max}\, S_{ic}^{(\ell)}. \tag{26}$$

For an isolated node in the original graph, its current label is retained. This exception is inactive in the evaluated largest connected components, which contain more than one node.

Synchronous reassignment does not enforce nonempty communities. No explicit empty-community repair is performed, so the final number of nonempty communities may be smaller than the input count C. The actual output count is recorded for each run. The fraction of changed labels is

$$r^{(\ell)} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left(y_i^{(\ell+1)} \neq y_i^{(\ell)}\right). \tag{27}$$

Refinement stops when $r^{(\ell)} \leq \tau$ or the maximum number of iterations $T_{\max}$ is reached. The iteration limit ensures finite termination; monotonic improvement of a common objective is not established for the penalized synchronous updates. The complete procedure is summarized in Algorithm A2.

3.5. Computational Complexity

Let N, e, and F denote the numbers of nodes, edges, and attributes in the retained largest connected component, respectively. Let r be the reduced dimension, C the number of communities, T the number of refinement iterations, and $K_{\max}$ the maximum propagation depth. The depth scan costs $O((K_{\max}+1)(e+N+m)F)$, where m is the total number of retained probe pairs. Density-peak initialization costs $O(N^2 r)$, and centroid-based refinement costs $O(TNCr)$. The implementation also computes all-pairs shortest paths, requiring $O(N^2)$ storage and up to $O(N^3)$ time. The principal scalability bottlenecks are therefore shortest-path computation and pairwise distance calculation.

4. Experiments and Analysis

This section evaluates the proposed framework on five benchmark attributed networks through comparisons with ten baseline methods, propagation-depth analysis, component ablations, and parameter-sensitivity experiments.

4.1. Datasets

Experiments are conducted on Cora, Citeseer, PubMed, ACM, and DBLP. Cora, Citeseer, and PubMed are citation networks obtained from the Planetoid repository https://github.com/kimiyoung/planetoid, (accessed on 28 November 2026), whereas ACM and DBLP are bibliographic network benchmarks distributed through PyDGC https://github.com/Marigoldwu/PyDGC, (accessed on 28 November 2026). The largest connected component (LCC) of each graph is retained before propagation and subsequent clustering. Table 1 reports the statistics of both the full input graphs and the retained LCCs, including the percentage of nodes retained.

Table 1. Statistics of the full input graphs and the largest connected components (LCCs) used in the experiments.

Dataset	Full Graph		LCC		Features	Classes	Retained (%)
	Nodes	Edges	Nodes	Edges	F	C	
Cora	2708	5278	2485	5069	1433	7	91.77
Citeseer	3327	4552	2120	3679	3703	6	63.72
PubMed	19,717	44,324	19,717	44,324	500	3	100.00
ACM	3025	13,128	2006	12,628	1870	3	66.31
DBLP	4057	3528	1711	2898	334	4	42.17

4.2. Baselines

We compare the proposed method with ten baseline methods organized into three groups: classical attribute clustering and topology-based community detection, deep attributed graph clustering, and adaptive graph propagation. These groups provide comparisons with methods using node attributes alone, network topology alone, or both sources of information.

Classical attribute and topology baselines.

- **K-means** is applied to the original node attributes after row-wise ℓ_2 normalization. It provides an attribute-only baseline without graph propagation.
- **Spectral Clustering (SC)** [41] partitions nodes using a spectral representation derived from the normalized graph Laplacian. It serves as a topology-only baseline.
- **Louvain** [14] detects communities through greedy modularity optimization. It uses network topology and produces a partition without requiring a target community count.
- **The Label Propagation Algorithm (LPA)** [15] is implemented using asynchronous updates, in which each node adopts a label occurring most frequently among its neighbors. It uses network topology and does not require a target community count.
- **Density Peak Clustering (DPC)** [35] provides a density-peak-based attribute clustering baseline. Our implementation selects centers using local density and separation information and assigns nodes to their nearest selected center. It uses row-normalized original attributes, without graph propagation or iterative community refinement. A scalable attribute-based approximation is used for PubMed.

Deep attributed graph clustering baselines.

- **Graph Autoencoder (GAE)** [27] learns node representations using a graph convolutional encoder and an inner-product decoder for graph reconstruction. K-means is applied to the learned representations to obtain the final partition.

- **Deep Attentional Embedded Graph Clustering (DAEGC)** [7] uses graph attention to learn node representations and jointly optimizes graph reconstruction and a self-training clustering objective.
- **Structural Deep Clustering Network (SDCN)** [29] integrates an autoencoder with a graph convolutional network through a delivery operator. Its dual self-supervised mechanism guides representation learning and clustering.

Adaptive graph propagation baselines.

- **Adaptive Graph Convolution (AGC)** [6] uses high-order graph convolution as a low-pass filter and selects the propagation order using intra-cluster distances obtained from intermediate spectral-clustering results. Our implementation uses a sparse affinity construction. AGC provides a related adaptive-propagation baseline whose depth selection relies on intermediate partitions.
- **GATFELPA** [10] is evaluated through an implemented adaptation combining Jaccard-based graph attention, similarity preservation, adaptive propagation-depth selection, and density-peak-guided enhanced label propagation. It provides a related baseline integrating adaptive propagation with density-based community detection.

The baseline methods were implemented following their published algorithmic descriptions and adapted to the common evaluation protocol. Attribute-based methods use row-wise ℓ_2-normalized features, whereas Louvain and the LPA use network topology only. All methods are evaluated on the same retained largest connected component and on the same nodes with valid reference labels.

For the reported runs, K-means uses 20 restarts and at most 300 iterations; GAE uses 100 training epochs; DAEGC and SDCN use 100 pretraining and 100 clustering epochs; AGC evaluates propagation orders up to 10 using a 64-dimensional SVD representation and a sparse 50-nearest-neighbour cosine affinity; DPC uses an 80-neighbour density estimate with a scalable procedure for larger graphs; and GATFELPA uses 64-dimensional representations and a maximum depth of 6. Louvain uses resolution 1.0, whereas the LPA uses asynchronous unweighted updates. Except for Louvain and the LPA, the methods receive the prescribed community count C; GAE uses C only for downstream K-means. In the main comparison, C equals the reference community count for each dataset. Louvain and the LPA are evaluated with their native output counts. The input count, where applicable, and the number of nonempty output communities C_{out} are recorded for each run.

4.3. Evaluation Metrics and Experimental Settings

Evaluation metrics. Performance is evaluated using accuracy (ACC), normalized mutual information (NMI), the adjusted Rand index (ARI), and the macro-averaged F1-score (F1).

For the n nodes with valid reference labels, let y_i and $\hat{y}_i$ denote the reference and predicted labels. ACC is computed after optimal one-to-one label matching:

$$\text{ACC} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{y_i = \pi^*(\hat{y}_i)\}, \tag{28}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function and π^* maximizes the number of correct assignments. When the community counts differ, unmatched predicted communities are mapped to dummy labels and contribute no correct assignments.

Using the same matching, F1 is averaged over the C_{ref} reference classes:

$$\text{F1} = \frac{1}{C_{\text{ref}}} \sum_{c=1}^{C_{\text{ref}}} \frac{2\text{TP}_c}{2\text{TP}_c + \text{FP}_c + \text{FN}_c}, \tag{29}$$

where TP_c, FP_c, and FN_c denote true positives, false positives, and false negatives. Unmatched reference classes and undefined terms receive zero F1.

NMI uses arithmetic-mean normalization, $NMI = 2I(Y;\widehat{Y})/[H(Y) + H(\widehat{Y})]$, where I denotes mutual information and H denotes entropy. The ARI measures partition agreement adjusted for chance. Both are computed without label matching. Higher values indicate better performance; ACC, NMI, and F1 lie in $[0, 1]$, whereas the ARI can be negative.

Experimental settings. For all five datasets, propagation depth is selected from $\{0, \ldots, 20\}$ using the label-free criterion in Section 3.2. Both low-similarity retention fractions are 0.20. Remote-pair construction uses a graph-distance quantile of 0.80, up to 50,000 candidate pairs, and at most 20,000 retained pairs. The SVD dimension and density-neighborhood size are $r = 64$ and $q = 80$, respectively. Refinement uses $\alpha = 0.8$, $\beta = 0.2$, at most 100 iterations, and a label-change tolerance of 0.0005.

Stochastic methods use ten seeds: $\{993, 550, 243, 16, 716, 383, 277, 274, 188, 796\}$. For the proposed method, these seeds control the randomized SVD, while the node pairs and selected depth remain fixed. DPC is deterministic and produces identical results across seeds. Performance scores and the numbers of nonempty communities in the retained LCC are reported as means and sample standard deviations.

4.4. Experimental Results

4.4.1. Comparison with Baseline Methods

We compare the proposed method with ten baselines on five attributed networks under the information conditions and experimental settings described in Sections 4.2 and 4.3. Tables 2–6 report performance scores and nonempty output-community counts (C_{out}) as means and sample standard deviations over ten seed settings. Bold values indicate the highest mean score for each metric. The proposed method ranks first in 15 of the 20 dataset–metric combinations by mean score and second in the remaining five. It leads on all four metrics for Citeseer, ACM, and DBLP, while AGC leads on Cora. On PubMed, it achieves the highest ACC, ARI, and F1 values, but slightly lower NMI than K-means.

Table 2. Comparison on Cora. Entries are mean ± sample standard deviation over ten seed settings.

Method	ACC	NMI	ARI	F1	C_{out}
K-means	0.3697 ± 0.0215	0.1900 ± 0.0348	0.1152 ± 0.0204	0.3402 ± 0.0321	7.00 ± 0.00
SC	0.4609 ± 0.0009	0.3943 ± 0.0019	0.2387 ± 0.0009	0.3201 ± 0.0172	7.00 ± 0.00
Louvain	0.4076 ± 0.0144	0.4628 ± 0.0049	0.2700 ± 0.0132	0.5528 ± 0.0094	27.20 ± 2.04
LPA	0.1417 ± 0.0142	0.4112 ± 0.0030	0.0626 ± 0.0095	0.2449 ± 0.0211	472.40 ± 7.75
DPC	0.4354 ± 0.0000	0.1858 ± 0.0000	0.1545 ± 0.0000	0.3795 ± 0.0000	7.00 ± 0.00
GAE	0.3656 ± 0.0520	0.2368 ± 0.0583	0.1376 ± 0.0413	0.3373 ± 0.0584	7.00 ± 0.00
DAEGC	0.5903 ± 0.0328	0.4601 ± 0.0377	0.3666 ± 0.0453	0.5514 ± 0.0337	7.00 ± 0.00
SDCN	0.3277 ± 0.0276	0.1331 ± 0.0415	0.0983 ± 0.0320	0.1657 ± 0.0136	4.00 ± 0.67
AGC	0.7497 ± 0.0012	0.5981 ± 0.0024	0.5428 ± 0.0024	0.7322 ± 0.0012	7.00 ± 0.00
GATFELPA	0.6181 ± 0.0256	0.4770 ± 0.0249	0.4125 ± 0.0307	0.5367 ± 0.0282	7.00 ± 0.00
Proposed	0.7363 ± 0.0012	0.5802 ± 0.0007	0.5170 ± 0.0025	0.7209 ± 0.0008	7.00 ± 0.00

Table 3. Comparison on Citeseer. Entries are mean ± sample standard deviation over ten seed settings.

Method	ACC	NMI	ARI	F1	C_{out}
K-means	0.5340 ± 0.0581	0.2840 ± 0.0440	0.2585 ± 0.0566	0.4862 ± 0.0554	6.00 ± 0.00
SC	0.5967 ± 0.0028	0.3711 ± 0.0020	0.3440 ± 0.0015	0.5058 ± 0.0050	6.00 ± 0.00
Louvain	0.2609 ± 0.0234	0.3246 ± 0.0040	0.1548 ± 0.0128	0.3534 ± 0.0246	35.90 ± 1.73
LPA	0.0935 ± 0.0148	0.3515 ± 0.0032	0.0398 ± 0.0100	0.1432 ± 0.0144	468.10 ± 12.36
DPC	0.3550 ± 0.0000	0.1293 ± 0.0000	0.1231 ± 0.0000	0.3101 ± 0.0000	6.00 ± 0.00
GAE	0.4511 ± 0.0624	0.2765 ± 0.0506	0.2331 ± 0.0560	0.4037 ± 0.0537	6.00 ± 0.00
DAEGC	0.5146 ± 0.0788	0.2937 ± 0.0605	0.2835 ± 0.0655	0.4631 ± 0.0807	6.00 ± 0.00
SDCN	0.3733 ± 0.0482	0.1478 ± 0.0548	0.1399 ± 0.0539	0.2342 ± 0.0398	4.10 ± 0.74
AGC	0.6259 ± 0.0012	0.4205 ± 0.0004	0.3824 ± 0.0010	0.5444 ± 0.0008	6.00 ± 0.00
GATFELPA	0.6378 ± 0.0132	0.4122 ± 0.0131	0.4076 ± 0.0159	0.5612 ± 0.0168	6.00 ± 0.00
Proposed	0.7027 ± 0.0015	0.4629 ± 0.0014	0.4699 ± 0.0023	0.6417 ± 0.0009	6.00 ± 0.00

Table 4. Comparison on PubMed. Entries are mean ± sample standard deviation over ten seed settings.

Method	ACC	NMI	ARI	F1	C_{out}
K-means	0.6011 ± 0.0001	0.3139 ± 0.0003	0.2813 ± 0.0002	0.5921 ± 0.0000	3.00 ± 0.00
SC	0.5089 ± 0.0016	0.1568 ± 0.0002	0.1125 ± 0.0002	0.4362 ± 0.0008	3.00 ± 0.00
Louvain	0.2255 ± 0.0241	0.1999 ± 0.0058	0.1012 ± 0.0110	0.3432 ± 0.0293	41.20 ± 3.12
LPA	0.0975 ± 0.0376	0.1787 ± 0.0026	0.0251 ± 0.0153	0.1467 ± 0.0439	2105.70 ± 47.94
DPC	0.4079 ± 0.0000	0.0643 ± 0.0000	0.0562 ± 0.0000	0.3931 ± 0.0000	3.00 ± 0.00
GAE	0.5007 ± 0.0597	0.0963 ± 0.0537	0.0684 ± 0.0614	0.4254 ± 0.1012	3.00 ± 0.00
DAEGC	0.6218 ± 0.0038	0.2082 ± 0.0040	0.1960 ± 0.0049	0.6248 ± 0.0038	3.00 ± 0.00
SDCN	0.5023 ± 0.0375	0.1903 ± 0.0244	0.1462 ± 0.0268	0.4357 ± 0.0687	3.00 ± 0.00
AGC	0.5169 ± 0.0001	0.1468 ± 0.0001	0.0944 ± 0.0000	0.4315 ± 0.0002	3.00 ± 0.00
GATFELPA	0.5239 ± 0.0519	0.1633 ± 0.0447	0.1479 ± 0.0557	0.5021 ± 0.0574	3.00 ± 0.00
Proposed	0.6732 ± 0.0001	0.3113 ± 0.0002	0.2952 ± 0.0001	0.6705 ± 0.0001	3.00 ± 0.00

Table 5. Comparison on ACM. Entries are mean ± sample standard deviation over ten seed settings.

Method	ACC	NMI	ARI	F1	C_{out}
K-means	0.8282 ± 0.0100	0.5202 ± 0.0155	0.5514 ± 0.0204	0.8342 ± 0.0093	3.00 ± 0.00
SC	0.8046 ± 0.0000	0.5009 ± 0.0000	0.5113 ± 0.0000	0.8113 ± 0.0000	3.00 ± 0.00
Louvain	0.2179 ± 0.0141	0.2975 ± 0.0061	0.1098 ± 0.0068	0.3526 ± 0.0164	33.20 ± 1.14
LPA	0.0846 ± 0.0136	0.2816 ± 0.0037	0.0263 ± 0.0035	0.1587 ± 0.0215	239.40 ± 6.02
DPC	0.4696 ± 0.0000	0.0572 ± 0.0000	0.0619 ± 0.0000	0.4676 ± 0.0000	3.00 ± 0.00
GAE	0.6668 ± 0.1146	0.3445 ± 0.1512	0.3357 ± 0.1413	0.6659 ± 0.1241	3.00 ± 0.00
DAEGC	0.8073 ± 0.0964	0.5498 ± 0.1231	0.5493 ± 0.1388	0.8115 ± 0.1047	3.00 ± 0.00
SDCN	0.6207 ± 0.0178	0.3942 ± 0.0833	0.3656 ± 0.0680	0.5285 ± 0.0450	2.50 ± 0.53
AGC	0.8774 ± 0.0000	0.6362 ± 0.0000	0.6638 ± 0.0000	0.8823 ± 0.0000	3.00 ± 0.00
GATFELPA	0.7275 ± 0.1063	0.4679 ± 0.1271	0.4399 ± 0.1531	0.7311 ± 0.1092	3.00 ± 0.00
Proposed	0.9073 ± 0.0000	0.7135 ± 0.0000	0.7360 ± 0.0000	0.9121 ± 0.0000	3.00 ± 0.00

Table 6. Comparison on DBLP. Entries are mean ± sample standard deviation over ten seed settings.

Method	ACC	NMI	ARI	F1	C_{out}
K-means	0.6634 ± 0.0065	0.3470 ± 0.0057	0.3168 ± 0.0112	0.6579 ± 0.0057	4.00 ± 0.00
SC	0.5123 ± 0.0009	0.2136 ± 0.0036	0.1800 ± 0.0003	0.3991 ± 0.0003	4.00 ± 0.00
Louvain	0.1613 ± 0.0097	0.2197 ± 0.0067	0.0735 ± 0.0051	0.2626 ± 0.0130	34.20 ± 1.03
LPA	0.0429 ± 0.0048	0.2940 ± 0.0023	0.0119 ± 0.0009	0.0821 ± 0.0091	394.10 ± 10.96
DPC	0.4354 ± 0.0000	0.1322 ± 0.0000	0.0657 ± 0.0000	0.4500 ± 0.0000	4.00 ± 0.00
GAE	0.4957 ± 0.0219	0.3017 ± 0.0243	0.2465 ± 0.0197	0.4860 ± 0.0295	4.00 ± 0.00
DAEGC	0.5967 ± 0.0445	0.3585 ± 0.0189	0.3509 ± 0.0292	0.5744 ± 0.0486	4.00 ± 0.00
SDCN	0.5930 ± 0.0461	0.3283 ± 0.0373	0.3364 ± 0.0675	0.4581 ± 0.1041	3.70 ± 0.67
AGC	0.7078 ± 0.0000	0.4296 ± 0.0000	0.4109 ± 0.0000	0.7050 ± 0.0000	4.00 ± 0.00
GATFELPA	0.6452 ± 0.0439	0.3815 ± 0.0290	0.3532 ± 0.0324	0.6203 ± 0.0591	4.00 ± 0.00
Proposed	0.8305 ± 0.0005	0.5684 ± 0.0006	0.6184 ± 0.0011	0.8224 ± 0.0004	4.00 ± 0.00

The proposed method returns $C_{out} = C$ in all 50 main runs, although synchronous refinement does not guarantee nonempty communities. SDCN produces fewer nonempty communities than the supplied count in some runs. Louvain and the LPA use topology alone without a prescribed community count, so their performance comparisons should be

interpreted under these different information conditions. Two-dimensional projections of the original attributes and compact representations are provided in Appendix B (Figure A1) to complement the quantitative results.

4.4.2. Validation of Adaptive Depth Selection

We evaluate the adaptive depth-selection criterion and its effect on the final clustering results. Table 7 summarizes the selected depths and corresponding pair-distance statistics, while Figure 2 shows the distance curves for all five networks. The selected depth K^* maximizes $S(k) = D_{\text{far}}(k) - D_{\text{near}}(k)$, with ties resolved in favor of the smallest depth. This procedure uses neither reference labels nor intermediate clustering.

Table 7. Propagation-depth selection on the retained LCCs.

Dataset	$\lvert\mathcal{P}_{\text{near}}\rvert$	$\lvert\mathcal{P}_{\text{far}}\rvert$	τ_{far}	K^*	$D_{\text{near}}(K^*)$	$D_{\text{far}}(K^*)$	$S(K^*)$
Cora	1014	2441	8	4	0.1058	0.7980	0.6922
Citeseer	736	2481	12	5	0.0848	0.8352	0.7504
PubMed	8865	2056	8	3	0.1373	0.8479	0.7106
ACM	2526	3273	7	1	0.0643	0.4501	0.3857
DBLP	580	2382	9	3	0.1217	0.8325	0.7108

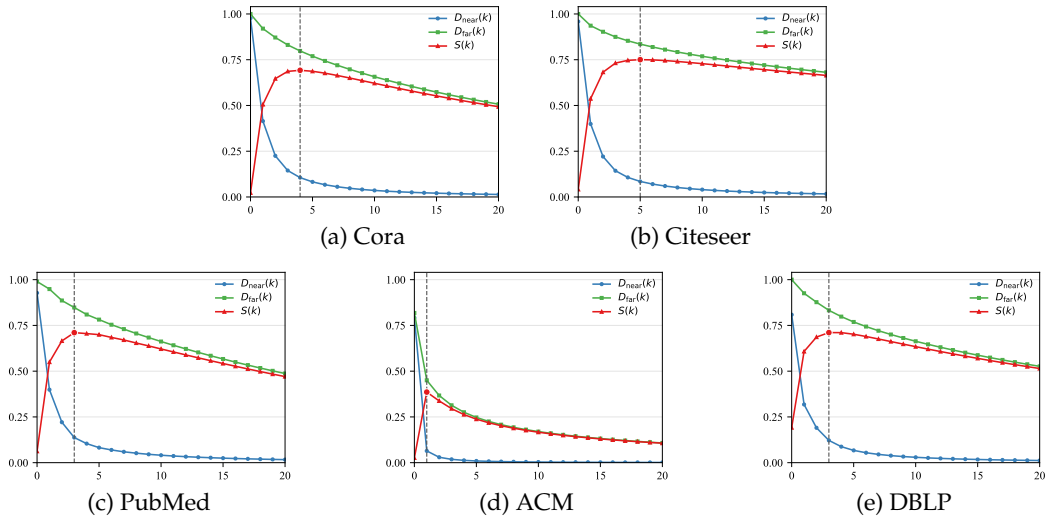

Figure 2. Propagation-depth selection on five attributed networks. Curves show the mean $D_{\text{near}}(k)$, $D_{\text{far}}(k)$, and $S(k)$; dashed lines mark K^* and symbols mark the maximum values.

The pair-distance contrast is a label-free proxy based on constructed adjacent and remote pairs, not on reference communities or a direct measure of community separation. Low-similarity adjacent pairs may cross community boundaries, and mean-distance differences do not capture distributional overlap. We therefore evaluate the complete clustering pipeline at every $k \in \{0, \ldots, 20\}$ while keeping the retained LCC, pair sets, all remaining parameters, and ten SVD seeds fixed. Figure 3 reports the resulting mean NMI and ARI value with sample standard deviations; reference labels are used only for post hoc evaluation and do not affect the selection of K^*.

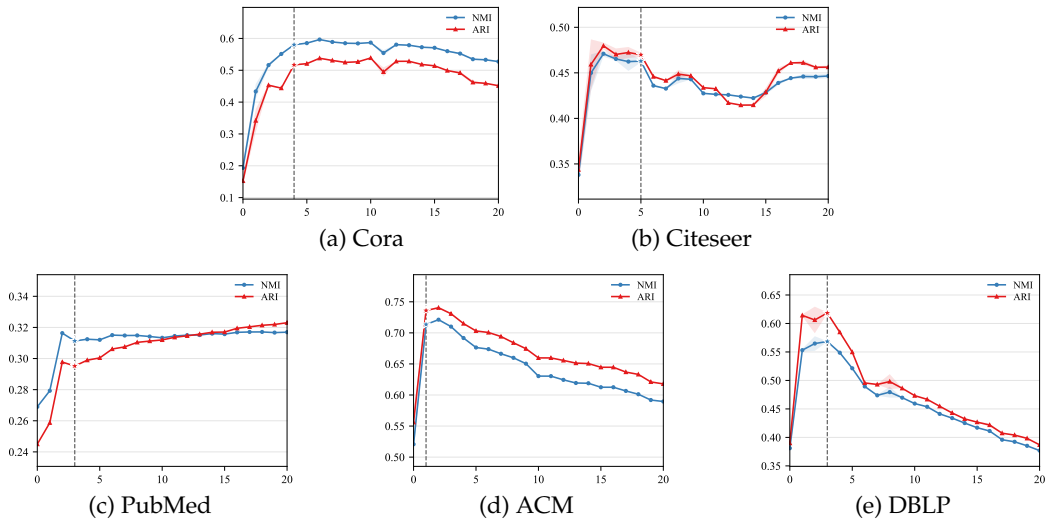

Figure 3. Clustering performance versus propagation depth. Curves show the mean NMI and ARI value, shaded bands indicate sample standard deviations, and dashed lines with stars mark the label-free selected depth K^*.

At K^*, both the mean NMI and ARI exceed their values at $k = 0$ on all five networks. On DBLP, $K^* = 3$ achieves the highest mean NMI and ARI value. On Cora, Citeseer, and ACM, the selected depths do not maximize either metric, with maximum gaps of 0.0163 for NMI and 0.0216 for the ARI. On PubMed, the selected depth gives NMI of 0.3113 and an ARI value of 0.2952, compared with maxima of 0.3171 and 0.3230 at larger depths. These results support the criterion as a practical label-free heuristic rather than a direct optimizer of external clustering metrics. The reported standard deviations reflect SVD randomness under fixed pair sets and do not assess robustness to independent pair resampling.

4.4.3. Ablation and Parameter Sensitivity

To examine the contributions of graph propagation and community refinement, Table 8 compares the full method with two variants under the same ten-seed protocol. The $K = 0$ variant retains SVD, density-peak initialization, and refinement but omits graph propagation. The K^* + DPC variant uses the selected propagation depth and density-peak initialization but omits refinement. It therefore differs from the standalone DPC baseline, which is applied directly to the original attributes.

Table 8. Component ablation on five attributed networks. Entries are mean ± sample standard deviation over ten seeds.

Dataset	$K = 0$		K^* + DPC		Full Method	
	NMI	ARI	NMI	ARI	NMI	ARI
Cora	0.1935 ± 0.0161	0.1523 ± 0.0114	0.4642 ± 0.0014	0.4219 ± 0.0013	0.5802 ± 0.0007	0.5170 ± 0.0025
Citeseer	0.3380 ± 0.0107	0.3437 ± 0.0160	0.4408 ± 0.0018	0.4429 ± 0.0035	0.4629 ± 0.0014	0.4699 ± 0.0023
PubMed	0.2690 ± 0.0004	0.2450 ± 0.0003	0.1852 ± 0.0010	0.1844 ± 0.0009	0.3113 ± 0.0002	0.2952 ± 0.0001
ACM	0.5207 ± 0.0122	0.5561 ± 0.0170	0.6768 ± 0.0026	0.6737 ± 0.0023	0.7135 ± 0.0000	0.7360 ± 0.0000
DBLP	0.3811 ± 0.0064	0.3899 ± 0.0057	0.4248 ± 0.0032	0.3281 ± 0.0026	0.5684 ± 0.0006	0.6184 ± 0.0011

The full method achieves higher mean NMI and ARI values than both variants on all five networks. The comparison with $K = 0$ indicates the benefit of graph propagation, whereas the comparison with K^* + DPC indicates that refinement improves the propagated initial partition. These results support the contributions of graph propagation and subsequent refinement under the reported settings.

We vary $\alpha \in \{0.6, 0.7, 0.8, 0.9, 1.0\}$ with $\beta = 1 - \alpha$, while keeping K^*, Z, and the initial partition fixed for each dataset and seed. Figure 4 reports mean NMI and ARI

values over ten seeds with sample standard deviations. The endpoint $\alpha = 1, \beta = 0$ removes the community-size penalty and yields the spherical K-means-type refinement described in Section 3.4 under the same initialization; it is therefore a matched ablation rather than an independently reinitialized baseline. Performance is dataset-dependent: Citeseer, PubMed, and DBLP favor $\alpha = 1$, ACM favors a lower value, and Cora is relatively flat near $\alpha = 0.8$–0.9. Accordingly, $\alpha = 0.8$ and $\beta = 0.2$ are used as common experimental settings rather than as a universally optimal or automatically selected value.

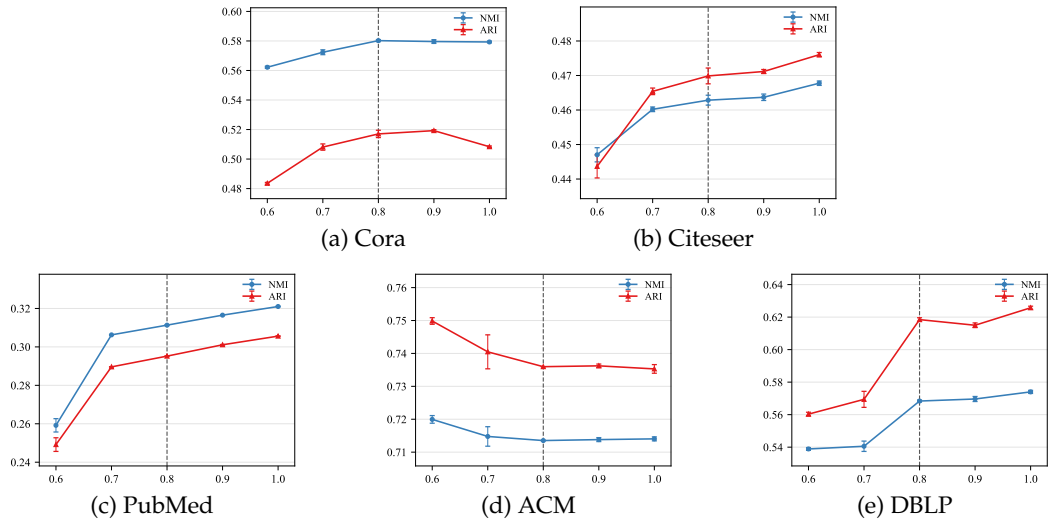

Figure 4. Sensitivity to the refinement weight α on five attributed networks, with $\beta = 1 - \alpha$. The rightmost endpoint $\alpha = 1$ therefore corresponds to $\beta = 0$, i.e., the no-size-penalty spherical K-means-type refinement under the fixed initial partition. Curves show mean NMI and ARI values over ten seeds; error bars indicate sample standard deviations. The dashed vertical line marks the common main-experiment setting $\alpha = 0.8$.

4.4.4. Robustness to Remote-Pair Sampling

We assess the sensitivity of the selected depth to the remote-pair candidate budget $B \in \{5000, 10{,}000, 20{,}000, 50{,}000\}$. For each dataset and budget, we perform 30 replicates. Within each replicate, all budgets draw independent subsets from a shared maximum candidate pool, and τ_{far} is recomputed for each subset. The retained LCC, hard-neighbor pairs, similarity fractions, candidate depth range $k \in \{0, \ldots, 20\}$, and all remaining settings are fixed. The pair sets remain unchanged during each depth scan, and reference labels and downstream clustering results are not used for pair construction or depth selection.

Table 9 reports the modal selected depth, its frequency, the mean $\pm$ sample standard deviation of K^* and $S(K^*)$, and the corresponding τ_{far}. Because the scan is performed on fixed propagated representations, the observed variation reflects remote-pair sampling rather than SVD randomness.

The modal selected depth remains unchanged across all budgets for Cora, Citeseer, PubMed, and ACM, with a frequency of 100%. For DBLP, the modal depth is also $K^* = 3$, while its selection frequency increases from 76.7% at $B = 5000$ to 96.7% at $B = 50{,}000$. The mean distance-contrast scores vary only slightly across budgets. These results support the stability of the criterion under the tested remote-pair sampling procedure and candidate budgets, but do not establish robustness to other pair-construction strategies, larger networks, or resampling procedures outside the tested candidate pools.

Table 9. Robustness of propagation-depth selection to remote-pair sampling. B denotes the candidate budget before filtering. Each budget uses an independent random subset within the maximum candidate pool of each replicate; each row summarizes 30 replicates.

Dataset	B	Modal K^*	Modal Frequency	K^*	$S(K^*)$	τ_{far}
Cora	5000	4	100.0%	4.000 ± 0.000	0.693 ± 0.005	8.00 ± 0.00
Cora	10,000	4	100.0%	4.000 ± 0.000	0.693 ± 0.003	8.00 ± 0.00
Cora	20,000	4	100.0%	4.000 ± 0.000	0.693 ± 0.002	8.00 ± 0.00
Cora	50,000	4	100.0%	4.000 ± 0.000	0.694 ± 0.002	8.00 ± 0.00
Citeseer	5000	5	100.0%	5.000 ± 0.000	0.751 ± 0.003	12.00 ± 0.00
Citeseer	10,000	5	100.0%	5.000 ± 0.000	0.752 ± 0.003	12.00 ± 0.00
Citeseer	20,000	5	100.0%	5.000 ± 0.000	0.751 ± 0.002	12.00 ± 0.00
Citeseer	50,000	5	100.0%	5.000 ± 0.000	0.752 ± 0.001	12.00 ± 0.00
PubMed	5000	3	100.0%	3.000 ± 0.000	0.706 ± 0.009	7.77 ± 0.43
PubMed	10,000	3	100.0%	3.000 ± 0.000	0.706 ± 0.008	7.80 ± 0.41
PubMed	20,000	3	100.0%	3.000 ± 0.000	0.708 ± 0.005	7.93 ± 0.25
PubMed	50,000	3	100.0%	3.000 ± 0.000	0.709 ± 0.003	7.97 ± 0.18
ACM	5000	1	100.0%	1.000 ± 0.000	0.384 ± 0.005	7.00 ± 0.00
ACM	10,000	1	100.0%	1.000 ± 0.000	0.387 ± 0.004	7.00 ± 0.00
ACM	20,000	1	100.0%	1.000 ± 0.000	0.386 ± 0.002	7.00 ± 0.00
ACM	50,000	1	100.0%	1.000 ± 0.000	0.386 ± 0.002	7.00 ± 0.00
DBLP	5000	3	76.7%	3.233 ± 0.430	0.712 ± 0.005	9.00 ± 0.00
DBLP	10,000	3	80.0%	3.200 ± 0.407	0.711 ± 0.003	9.00 ± 0.00
DBLP	20,000	3	86.7%	3.133 ± 0.346	0.710 ± 0.003	9.00 ± 0.00
DBLP	50,000	3	96.7%	3.033 ± 0.183	0.711 ± 0.002	9.00 ± 0.00

4.4.5. Sensitivity to the Prescribed Community Count

To assess sensitivity to the prescribed community count, we vary $C_{\text{in}} \in \{\max(2,C\text{-}2), C-1, C, C+1, C+2\}$ for each dataset, where C is the reference community count. The lower bound of two reflects the requirement of density-peak initialization. The tested ranges are 5–9 for Cora, 4–8 for Citeseer, 2–5 for PubMed and ACM, and 2–6 for DBLP.

For each C_{in}, the retained LCC, selected propagation depth, SVD representation, refinement weights $(\alpha, \beta) = (0.8, 0.2)$, stopping rule, and ten random seeds are kept unchanged, while density-peak centers and subsequent refinement are recomputed. Reference labels are used only for post hoc evaluation. Table 10 reports the ACC, NMI, ARI, macro-F1, and the number of nonempty output communities C_{out} as means and sample standard deviations over the ten seeds.

At the reference count $C_{\text{in}} = C$, the results reproduce the main-experiment values. All four metrics attain their highest mean at $C_{\text{in}} = C$ for Cora, PubMed, ACM, and DBLP. Citeseer is a small exception: ACC is highest at $C_{\text{in}} = 5$, whereas NMI, ARI, and F1 values are highest at $C_{\text{in}} = 6$. Counts below or above the reference value generally reduce performance. Across all 230 runs, $C_{\text{out}} = C_{\text{in}}$; this is an empirical result under the tested settings and does not guarantee nonempty communities for general synchronous refinement.

These results quantify sensitivity to community-count misspecification but do not estimate the count when it is unavailable. Louvain and the LPA remain count-free baselines and therefore operate under different community-count information conditions.

Table 10. Sensitivity to the prescribed input community count. Entries are means ± sample standard deviations over ten fixed seeds; C_{out} is the number of nonempty output communities.

Dataset	C_{in}	ACC	NMI	ARI	F1	C_{out}
Cora	5	0.6801 ± 0.0009	0.5532 ± 0.0021	0.4883 ± 0.0013	0.5196 ± 0.0008	5.00 ± 0.00
Cora	6	0.7270 ± 0.0169	0.5750 ± 0.0101	0.5001 ± 0.0010	0.6432 ± 0.0197	6.00 ± 0.00
Cora	7	0.7363 ± 0.0012	0.5802 ± 0.0007	0.5170 ± 0.0025	0.7209 ± 0.0008	7.00 ± 0.00
Cora	8	0.7031 ± 0.0119	0.5716 ± 0.0025	0.4898 ± 0.0056	0.6155 ± 0.0033	8.00 ± 0.00
Cora	9	0.6196 ± 0.0188	0.5536 ± 0.0072	0.4234 ± 0.0231	0.5239 ± 0.0095	9.00 ± 0.00
Citeseer	4	0.5944 ± 0.0419	0.4109 ± 0.0246	0.3959 ± 0.0322	0.4487 ± 0.0269	4.00 ± 0.00
Citeseer	5	0.7129 ± 0.0050	0.4569 ± 0.0072	0.4670 ± 0.0064	0.6050 ± 0.0050	5.00 ± 0.00
Citeseer	6	0.7027 ± 0.0015	0.4629 ± 0.0014	0.4699 ± 0.0023	0.6417 ± 0.0009	6.00 ± 0.00
Citeseer	7	0.6029 ± 0.0254	0.4418 ± 0.0066	0.4107 ± 0.0193	0.5015 ± 0.0117	7.00 ± 0.00
Citeseer	8	0.5525 ± 0.0062	0.4255 ± 0.0024	0.3728 ± 0.0058	0.4200 ± 0.0046	8.00 ± 0.00
PubMed	2	0.5646 ± 0.0000	0.1399 ± 0.0001	0.0995 ± 0.0000	0.4130 ± 0.0000	2.00 ± 0.00
PubMed	3	0.6732 ± 0.0001	0.3113 ± 0.0002	0.2952 ± 0.0001	0.6705 ± 0.0001	3.00 ± 0.00
PubMed	4	0.5523 ± 0.0001	0.2571 ± 0.0001	0.2158 ± 0.0001	0.4503 ± 0.0001	4.00 ± 0.00
PubMed	5	0.4420 ± 0.0061	0.2395 ± 0.0025	0.1754 ± 0.0028	0.3329 ± 0.0039	5.00 ± 0.00
ACM	2	0.6227 ± 0.0002	0.5904 ± 0.0007	0.4650 ± 0.0002	0.5400 ± 0.0002	2.00 ± 0.00
ACM	3	0.9073 ± 0.0000	0.7135 ± 0.0000	0.7360 ± 0.0000	0.9121 ± 0.0000	3.00 ± 0.00
ACM	4	0.7333 ± 0.0095	0.6291 ± 0.0083	0.6180 ± 0.0093	0.6130 ± 0.0034	4.00 ± 0.00
ACM	5	0.6646 ± 0.0419	0.5951 ± 0.0091	0.5644 ± 0.0275	0.4629 ± 0.0153	5.00 ± 0.00
DBLP	2	0.5757 ± 0.0002	0.3804 ± 0.0004	0.3755 ± 0.0003	0.3537 ± 0.0001	2.00 ± 0.00
DBLP	3	0.7251 ± 0.0010	0.5298 ± 0.0026	0.5621 ± 0.0023	0.5945 ± 0.0008	3.00 ± 0.00
DBLP	4	0.8305 ± 0.0005	0.5684 ± 0.0006	0.6184 ± 0.0011	0.8224 ± 0.0004	4.00 ± 0.00
DBLP	5	0.6680 ± 0.0059	0.5139 ± 0.0020	0.4867 ± 0.0087	0.5759 ± 0.0041	5.00 ± 0.00
DBLP	6	0.5811 ± 0.0114	0.4930 ± 0.0034	0.4341 ± 0.0091	0.4445 ± 0.0058	6.00 ± 0.00

4.4.6. Runtime, Memory, and Empirical Scalability

We measure wall-clock time and peak resident memory under the same retained-LCC protocol and parameter settings as the main experiments. Pair construction is timed at the main candidate budget $B = 50{,}000$, including candidate-pool generation and pair filtering, and is averaged over the 30 replicates in Section 4.4.4. Detector-stage time is taken from the ten fixed seeds in Section 4.4.1 after the pair sets and selected depth have been fixed, and is reported as the mean ± sample standard deviation. The reported total is the sum of these two means, rather than a separately timed end-to-end run. Peak RSS is measured in an isolated detector process with seed 993. Reference labels are used only for post hoc evaluation. Measurements were collected on a 64-bit Windows 10 system with an Intel CPU and 12 logical processors, using Python 3.10.20, NumPy 2.2.6, SciPy 1.15.2, and scikit-learn 1.7.2.

To identify the dominant computational stages, we performed one isolated profiling run per dataset using seed 993, after the pair sets and selected depth had been fixed. These diagnostic stage times were measured separately from the ten-seed end-to-end timings and were not used to calculate the detector or total times in Table 11. The results are shown in Table 12.

Table 11. Runtime and peak detector memory on the retained largest connected components. Pair construction is averaged over the 30 replicates at $B = 50{,}000$. Detector time is the end-to-end mean ± sample standard deviation over the ten main seeds. The reported total is the sum of the pair-construction and detector-stage means. Peak RSS and stage-wise diagnostics are measured separately.

Dataset	Nodes	Edges	Pair (s)	Detector (s)	Total (s)	Peak RSS (MB)
Cora	2485	5069	3.46	0.81 ± 0.13	4.28	235.34
Citeseer	2120	3679	5.37	1.13 ± 0.15	6.50	315.50
PubMed	19,717	44,324	65.02	44.18 ± 0.23	109.20	468.41
ACM	2006	12,628	5.81	0.46 ± 0.02	6.27	219.18
DBLP	1711	2898	0.95	0.24 ± 0.01	1.20	165.12

Table 12. Stage-wise runtime and peak RSS from one isolated profiling run per dataset using seed 993.

Dataset	Load (s)	Propag. (s)	SVD (s)	DPC (s)	Refine. (s)	RSS (MB)
Cora	0.20	0.11	0.66	0.19	0.03	235.34
Citeseer	0.10	0.18	0.79	0.14	0.02	315.50
PubMed	0.13	0.51	1.86	50.48	0.07	468.41
ACM	0.06	0.06	0.26	0.12	0.00	219.18
DBLP	0.05	0.02	0.11	0.10	0.01	165.12

The isolated profile indicates that density-peak initialization dominates the PubMed detector-stage computation, whereas propagation, SVD, and refinement are comparatively smaller.

5. Discussion

The proposed framework integrates fixed topology–attribute probe pairs, label-free propagation-depth selection, density-peak initialization, and centroid-based refinement without neural-network parameter training. Across five benchmark networks and ten baseline methods, it ranks first in 15 of the 20 dataset–metric combinations and second in the remaining five. It performs best on all four metrics for Citeseer, ACM, and DBLP, whereas AGC performs better on Cora and *K*-means obtains a slightly higher NMI on PubMed. The component ablation further supports the contributions of graph propagation and subsequent refinement under the reported settings.

The pair-distance criterion selects the propagation depth without ground-truth labels or intermediate clustering. The selected depth is optimal or close to the best tested value on Cora, Citeseer, ACM, and DBLP, but the criterion does not maximize every external metric on PubMed. Pair-sampling experiments show stable depth selection under the tested candidate budgets, although DBLP benefits from larger budgets. The weight-sensitivity results are dataset-dependent; therefore, $\alpha = 0.8$ and $\beta = 0.2$ are used as shared experimental settings, rather than as a universally optimal or automatically selected rule. Figure 4 includes the matched $\alpha = 1, \beta = 0$ refinement case without the community-size penalty, using the same propagated representation and initial partition. A fully independent spherical *K*-means baseline with separately controlled initialization would provide an additional comparison.

The method requires a prescribed community count and is evaluated on the largest connected component. The count-sensitivity experiment shows that performance depends on this input, while Louvain and the LPA provide count-free baselines under different information conditions. The selected node pairs are heuristic probes whose correspondence to true within-community and between-community pairs is not directly verified. Runtime profiling also identifies pair construction and density-peak initialization as the main computational costs on the largest tested network. Future work should investigate community-count estimation without reference labels, evaluation over all graph components, scalable pair construction and density estimation, and independently initialized spherical *K*-means comparisons.

6. Conclusions

This study presents a training-free framework for community detection in attributed networks with a prescribed community count. The method selects a propagation depth using a label-free distance contrast over topology–attribute-filtered node pairs and then combines density-peak initialization with centroid-based refinement.

The reported evaluation is restricted to the retained largest connected component of each dataset and uses a prescribed community count. Under this protocol, the proposed method ranks first in 15 of the 20 dataset–metric combinations and second in the remaining five. It performs best on all four metrics for Citeseer, ACM, and DBLP, whereas AGC performs better on Cora and K-means obtains a slightly higher NMI on PubMed. The ablation and sensitivity analyses indicate benefits from graph propagation and refinement under the tested settings, while showing that the depth criterion is heuristic and that the preferred refinement weight is dataset-dependent. Accordingly, $\alpha = 0.8$ and $\beta = 0.2$ are used as shared experimental settings rather than as an automatically selected rule.

These findings are limited to the retained largest connected component of each dataset and the prescribed-count protocol. The count-sensitivity results show that performance depends on the supplied count, while Louvain and the LPA provide count-free baselines under different information conditions. The matched $\alpha = 1, \beta = 0$ endpoint in Figure 4 provides a no-size-penalty refinement comparison under the same initialization. An independently initialized spherical K-means baseline remains a possible extension. Future work should investigate scalable pair construction and density estimation, evaluation over all graph components, and community-count estimation without reference labels.

Author Contributions: T.Z.: methodology, software, and writing; X.R.: data preparation, reviewing, and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Doctoral Research Startup Fund for introduced talents of Lanzhou University of Technology (Grant No. 14/062203).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Algorithmic Procedure

Algorithm A1 selects the propagation depth and constructs the compact representation. Algorithm A2 then uses this representation for density-peak initialization and centroid-based refinement.

Algorithm A1 Adaptive propagation-depth selection and compact representation

Input: Attributed graph $G = (V, E, X)$; maximum depth $K_{\max}$; pair-selection settings; SVD rank r.
Output: Selected depth K^* and compact representation Z.

Row-normalize X to obtain X_0; form $P = \widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}$.
Construct the fixed hard-neighbor and structurally remote pair sets $\mathcal{P}_{\text{near}}$ and $\mathcal{P}_{\text{far}}$ from G and X_0.
for $k = 0, 1, \ldots, K_{\max}$ **do**
    Compute $H^{(k)} = P^k X_0$ and row-normalize it to $\widehat{H}^{(k)}$.
    Evaluate $D_{\text{near}}(k)$, $D_{\text{far}}(k)$, and $S(k) = D_{\text{far}}(k) - D_{\text{near}}(k)$ on the fixed pairs.
end for
Set K^* to the smallest depth attaining $\max_{0 \le k \le K_{\max}} S(k)$.
Set $\widetilde{X} = \widehat{H}^{(K^*)}$; compute the rank-$r$ truncated SVD $\widetilde{X} \approx U_r \Sigma_r V_r^\top$ and row-normalize $Z = U_r \Sigma_r$.
return (K^*, Z).

Algorithm A2 Density-peak initialization and centroid-based refinement

Input: Graph G; compact representation Z from Algorithm A1; community count C; density-neighbor count q; refinement weights α, β; stopping tolerance τ; iteration limit $T_{\max}$.
Output: Community labels y.

Compute cosine dissimilarities δ_{ij} in Z; estimate local densities ρ_i, relative distances Δ_i, and peak scores γ_i.
Select the highest-density node and the $C-1$ remaining nodes with largest γ_i as centers; assign every node to its nearest center to obtain $y^{(0)}$.
for $\ell = 0, 1, \ldots, T_{\max} - 1$ **do**
    Compute community sizes $m_c^{(\ell)}$ and row-normalized centroids $\widehat{\mu}_c^{(\ell)}$ from $y^{(\ell)}$.
    Compute each node's centroid similarities and standardize them to obtain $\widehat{R}_{ic}^{(\ell)}$.
    Form $S_{ic}^{(\ell)} = \alpha \widehat{R}_{ic}^{(\ell)} - \beta \log(m_c^{(\ell)} + 1)$; synchronously assign each non-isolated node to its highest-scoring community.
    Keep isolated-node labels unchanged and compute the label-change ratio $r^{(\ell)}$.
    if $r^{(\ell)} \leq \tau$ **then**
        break
    end if
end for
return the final labels y.

Appendix B. Illustrative Results

Figure A1 visualizes the attribute space and the final compact representation from one saved run on each of the five benchmark networks. For each dataset, the same 1500 nodes were randomly sampled from its largest connected component with seed 42. The first row displays original row-normalized attributes after a 40-dimensional truncated-SVD reduction; the second and third rows display the final compact representation $\mathbf{Z}$. Each representation was projected to two dimensions with t-SNE using the same settings. The lower two rows share identical coordinates, making their reference-label and predicted-label colors directly comparable. Predicted community IDs were matched to reference IDs only for display; this matching was not used during propagation, clustering, or refinement. Since t-SNE does not preserve all high-dimensional distances, these plots illustrate the partitions rather than replace the quantitative evaluation in Section 4.

The shared coordinates in the lower two rows make local agreement and disagreement between the predicted and reference partitions visible. Some colors remain intermingled, particularly on PubMed; the projection should therefore not be read as evidence of perfect community separation.

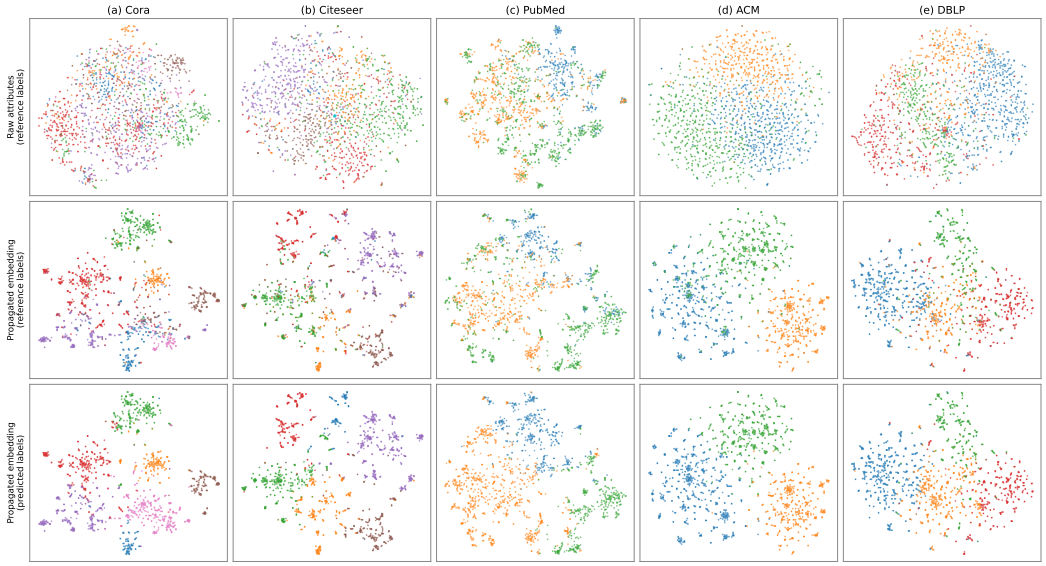

Figure A1. Two-dimensional visualization of 1500 sampled nodes per dataset: original attributes colored by reference labels (**top**), final propagated compact representation colored by reference labels (**middle**), and the same compact representation colored by the proposed method's predicted labels (**bottom**). Different colors denote different communities within each dataset. Predicted label colors are matched to reference colors solely for visual comparison.

References

1. Fortunato, S. Community detection in graphs. *Phys. Rep*. **2010**, *486*, 75–174.
2. Jin, D.; Yu, Z.; Jiao, P.; Pan, S.; He, D.; Wu, J.; Yu, P.S.; Zhang, W. A survey of community detection approaches: From statistical modeling to deep learning. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 1149–1170.
3. Chunaev, P. Community detection in node-attributed social networks: A survey. *Comput. Sci. Rev.* **2020**, *37*, 100286.
4. Yang, J.; McAuley, J.; Leskovec, J. Community detection in networks with node attributes. In Proceedings of the 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, 7–10 December 2013; pp. 1151–1156.
5. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. In Proceedings of the 5th International Conference on Learning Representations (ICLR), Toulon, France, 24–26 April 2017.
6. Zhang, X.; Liu, H.; Li, Q.; Wu, X.-M. Attributed graph clustering via adaptive graph convolution. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI), Macao, China, 10–16 August 2019; pp. 4327–4333.
7. Wang, C.; Pan, S.; Hu, R.; Long, G.; Jiang, J.; Zhang, C. Attributed graph clustering: A deep attentional embedding approach. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI), Macao, China, 10–16 August 2019; pp. 3670–3676.
8. Li, Q.; Han, Z.; Wu, X.-M. Deeper insights into graph convolutional networks for semi-supervised learning. *Proc. AAAI Conf. Artif. Intell.* **2018**, *32*, 3538–3545.
9. Chen, D.; Lin, Y.; Li, W.; Li, P.; Zhou, J.; Sun, X. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 3438–3445.
10. Tang, F.; Li, J.; Liu, X.; Chang, C.; Teng, L. GATFELPA integrates graph attention networks and enhanced label propagation for robust community detection. *Sci. Rep.* **2025**, *15*, 3952.
11. Newman, M.E.J.; Girvan, M. Finding and evaluating community structure in networks. *Phys. Rev. E* **2004**, *69*, 026113.
12. Newman, M.E.J. Finding community structure using the eigenvectors of matrices. *Phys. Rev. E* **2006**, *74*, 036104.
13. Clauset, A.; Newman, M.E.J.; Moore, C. Finding community structure in very large networks. *Phys. Rev. E* **2004**, *70*, 066111.
14. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**, *2008*, P10008.
15. Raghavan, U.N.; Albert, R.; Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E* **2007**, *76*, 036106.
16. Rosvall, M.; Bergstrom, C.T. Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci. USA* **2008**, *105*, 1118–1123.
17. Holland, P.W.; Laskey, K.B.; Leinhardt, S. Stochastic blockmodels: First steps. *Soc. Netw.* **1983**, *5*, 109–137.
18. Zhou, Y.; Cheng, H.; Yu, J.X. Graph clustering based on structural/attribute similarities. *Proc. VLDB Endow.* **2009**, *2*, 718–729.

19. Zhou, Y.; Cheng, H.; Yu, J.X. Clustering large attributed graphs: An efficient incremental approach. In Proceedings of the 2010 IEEE 10th International Conference on Data Mining (ICDM), Sydney, Australia, 13–17 December 2010; pp. 689–698.
20. Ruan, Y.; Fuhry, D.; Parthasarathy, S. Efficient community detection in large networks using content and links. In Proceedings of the 22nd International Conference on World Wide Web, Rio de Janeiro, Brazil, 13–17 May 2013; pp. 1089–1098.
21. Xu, Z.; Ke, Y.; Wang, Y.; Cheng, H.; Cheng, J. A model-based approach to attributed graph clustering. In Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, Scottsdale, AZ, USA, 20–24 May 2012; pp. 505–516.
22. Wang, X.; Jin, D.; Cao, X.; Yang, L.; Zhang, W. Semantic community identification in large attribute networks. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 265–271.
23. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710.
24. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
25. Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; Mei, Q. LINE: Large-scale information network embedding. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 1067–1077.
26. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph attention networks. In Proceedings of the 6th International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
27. Kipf, T.N.; Welling, M. Variational graph auto-encoders. *arXiv* **2016**, arXiv:1611.07308.
28. Pan, S.; Hu, R.; Long, G.; Jiang, J.; Yao, L.; Zhang, C. Adversarially regularized graph autoencoder for graph embedding. In Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 2609–2615.
29. Bo, D.; Wang, X.; Shi, C.; Zhu, H.; Lu, E.; Cui, P. Structural deep clustering network. In Proceedings of the Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 1400–1410.
30. Oono, K.; Suzuki, T. Graph neural networks exponentially lose expressive power for node classification. In Proceedings of the 8th International Conference on Learning Representations (ICLR), Online, 26–30 April 2020.
31. Chen, M.; Wei, Z.; Huang, Z.; Ding, B.; Li, Y. Simple and deep graph convolutional networks. In Proceedings of the 37th International Conference on Machine Learning (ICML), Online, 13–18 July 2020; pp. 1725–1735.
32. Rong, Y.; Huang, W.; Xu, T.; Huang, J. DropEdge: Towards deep graph convolutional networks on node classification. In Proceedings of the 8th International Conference on Learning Representations (ICLR), Online, 26–30 April 2020.
33. Zhao, L.; Akoglu, L. PairNorm: Tackling oversmoothing in GNNs. In Proceedings of the 8th International Conference on Learning Representations (ICLR), Online, 26–30 April 2020.
34. Chien, E.; Peng, J.; Li, P.; Milenkovic, O. Adaptive universal generalized PageRank graph neural networks. In Proceedings of the 9th International Conference on Learning Representations (ICLR), Online, 3–7 May 2021.
35. Rodriguez, A.; Laio, A. Clustering by fast search and find of density peaks. *Science* **2014**, *344*, 1492–1496.
36. Chen, L.; Zheng, H.; Li, Y.; Liu, Z.; Zhao, L.; Tang, H. Enhanced density peak-based community detection algorithm. *J. Intell. Inf. Syst.* **2022**, *59*, 263–284.
37. Lu, H.; Sang, X.; Zhao, Q.; Lu, J. Community detection algorithm based on nonnegative matrix factorization and improved density peak clustering. *IEEE Access* **2020**, *8*, 5749–5759.
38. Jia, Y.; Ouyang, L.; Wang, Q.; Li, H. Overlapping community detection on attributed graphs via neutrosophic C-means. *Chin. Phys. B* **2025**, *34*, 128901.
39. Gao, H.; Han, J.; Liu, Y.; Zhang, P.; Yang, B.; Zu, Y.; Liu, F.; Qian, Y. K-means community detection algorithm based on density peaks. *Entropy* **2026**, *28*, 152.
40. Dhillon, I.S.; Modha, D.S. Concept decompositions for large sparse text data using clustering. *Mach. Learn.* **2001**, *42*, 143–175.
41. Shi, J.; Malik, J. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 888–905.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.