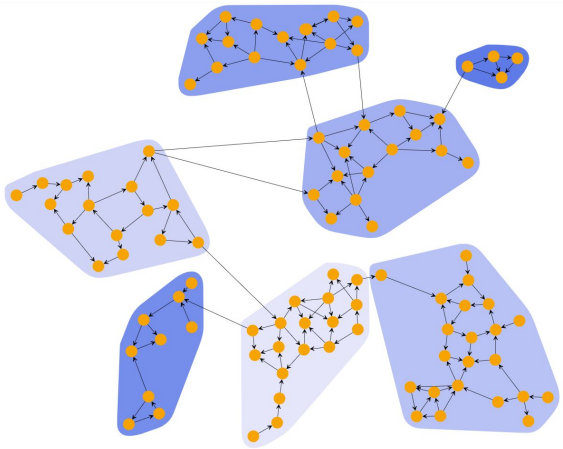
Multi-view Contrastive Graph Clustering

What is Graph Clustering?

Definition: Graph clustering divides unlabeled nodes of graph into clusters.

Typical methodology: Learn a good representation of the graph -> apply classical clustering method upon embeddings.

Common methods: LINE, GAE, etc.



Motivation for multi-view clustering

Real-life data:

- Collected from different sources -> represented by different features (e.g., for image data, color & texture info can be two different features; for movie network, co-directors & co-actors can generate different views of the graph)
- Each view can be noisy and incomplete
- Tend to share similar geometry & semantics -> complementary data

Better performance overall if we can integrate all the views!

Some existing methods

- Multi-view Spectral Clustering Network
  - uses a deep learning network to enforce within-view similarity and between-view consistency.
  - cons: developed for feature matrix, cannot handle graph data.
- Principled multilayer network embedding
  - projects a multilayer network into a continuous vector space.
  - cons: fail to explore the feature information.
- Multi-view attribute GCNs for clustering
  - exploits information of all views and adopts a cross-view consensus learning by enforcing the representations of different views to be as similar as possible.
  - cons: not directly applicable to multiple graphs data with multi-view attributes.

Multi-view Contrastive Graph Clustering (MCGC)

- Goal: generate clustering on attributed graph with multi-view features and multiple topological graphs.
- General steps:
  a. Learn a smooth signals via graph filtering
  b. Learn an optimized graph shared by all views from the smooth representations
  c. Utilize contrastive loss at graph-level to learn a more discriminative consensus graph

Notation

V: set of N nodes

A_v: normalized adjacency matrices under different views

D_v: degree matrices under different views.

L_v: normalized graph laplacian (L_v = I - A_v)

X_v: feature matrices under different views

Graph Filtering

$$\min_{H} \|H - X\|_F^2 + s \operatorname{Tr}\left(\mathrm{H}^\top \mathrm{LH}\right)$$

$$H = (I + sL)^{-1} X.$$

Goal:

- Learn smooth representations of the feature matrices.
- Filter out high-frequency noise while preserving the graph geometric features.

Graph Learning

$$\min_{S} \left\| H^{\top} - H^{\top} S \right\|_F^2 + \alpha \|S\|_F^2$$

$$\min_{S,\lambda^v} \sum_{v=1}^{V} \lambda^v \left(\left\| H^{v\top} - H^{v\top} S \right\|_F^2 + \alpha \|S\|_F^2 \right) + \sum_{v=1}^{V} (\lambda^v)^{\gamma}$$

Goal:

- Learn an consensus graph shared by all views
- Learn the graph from smooth representation to filter noisy & incomplete data

Why it works: Data's self-expression property: each data point can be represented by a linear combination of other data samples

Graph Contrastive Regularizer

$$\mathcal{J} = \sum_{i=1}^{N} \sum_{j \in \mathbb{N}_i^v} -\log \frac{\exp\left(S_{ij}\right)}{\sum_{p \neq i}^{N} \exp\left(S_{ip}\right)},$$

Goal:

- Draw neighbors close and push non-neighbors apart

Difference with instant-level contrastive learning:

- Positive samples: each node and its kNNs (rather than corrupted nodes)
- Contrastive regularizer on graph matrix S rather than on node features.

Loss function

$$\min_{S,\lambda^v} \sum_{v=1}^{V} \lambda^v \left(\left\| H^{v\top} - H^{v\top} S \right\|_F^2 + \alpha \sum_{i=1}^{N} \sum_{j \in \mathbb{N}_i^v} -\log \frac{\exp\left(S_{ij}\right)}{\sum_{p \neq i}^{N} \exp\left(S_{ip}\right)} \right) + \sum_{v=1}^{V} \left(\lambda^v\right)^{\gamma}$$

Datasets & Metrics

Table 1: The statistical information of datasets.

Dataset	Nodes	Features	Graph and Edges	Clusters
ACM	3,025	1,830	Co-Subject (29,281) Co-Author (2,210,761)	3
DBLP	4,057	334	Co-Author (11,113) Co-Conference (5,000,495) Co-Term (6,776,335)	4
IMDB	4,780	1,232	Co-Actor (98,010) Co-Director (21,018)	3
Amazon photos	7,487	745 7,487	Co-Purchase(119,043)	8
Amazon computers	13,381	767 13,381	Co-Purchase(245,778)	10

Metrics: Accuracy, Normalized Mutual Information, Adjusted Rand Index, F1 score

Result

Table 2: Results on ACM, DBLP, IMDB.

Method		LINE	GAE	PMNE	RMSC	SwMC	O2MA	O2MAC	MCGC	MCGC*
ACM	ACC	0.6479	0.8216	0.6936	0.6315	0.3831	0.888	0.9042	**0.9147**	0.9055
	NMI	0.3941	0.4914	0.4648	0.3973	0.4709	0.6515	0.6923	**0.7126**	0.6823
	ARI	0.3433	0.5444	0.4302	0.3312	0.0838	0.6987	0.7394	**0.7627**	0.7385
	F1	0.6594	0.8225	0.6955	0.5746	0.018	0.8894	0.9053	**0.9155**	0.9062
DBLP	ACC	0.8689	0.8859	0.7925	0.8994	0.3253	0.904	0.9074	**0.9298**	0.9162
	NMI	0.6676	0.6925	0.5914	0.7111	0.019	0.7257	0.7287	**0.8302**	0.7490
	ARI	0.6988	0.741	0.5265	0.7647	0.0159	0.7705	0.778	0.7746	**0.7995**
	F1	0.8546	0.8743	0.7966	0.8248	0.2808	0.8976	0.9013	**0.9252**	0.9112
IMDB	ACC	0.4268	0.4298	0.4958	0.2702	0.2453	0.4697	0.4502	**0.6182**	0.6113
	NMI	0.0031	0.0402	0.0359	0.3775	0.0023	0.0524	0.0421	0.1149	**0.1225**
	ARI	-0.009	0.0473	0.0366	0.0054	0.0017	0.0753	0.0564	**0.1833**	0.1811
	F1	0.287	0.4062	0.3906	0.0018	0.3164	0.4229	0.1459	0.4401	**0.4512**

Table 3: Results on Amazon photos and Amazon computers. The '-' means that the method raises out-of-memory problem.

Dataset	Amazon photos				Amazon computers			
	ACC	NMI	ARI	F1	ACC	NMI	ARI	F1
COMPLETER	0.3678	0.2606	0.0759	0.3067	0.2417	0.1562	0.0536	0.1601
MVGRL	0.5054	0.4331	0.2379	0.4599	0.2450	0.1012	0.0553	0.1706
MAGCN	0.5167	0.3897	0.2401	0.4736	–	–	–	–
MCGC	**0.7164**	**0.6154**	**0.4323**	**0.6864**	**0.5967**	**0.5317**	**0.3902**	**0.5204**

GCMC*: only use the shared neighbors in the contrastive loss

Contribution

- Novel contrastive loss at graph-level
- A clustering framework that can handle multilayer graphs with multi-view attributes.
- Achieves state-of-the-art performance

Potential limitation

- Take up lots of memory: the size of learned graph is of size N x N
- Graph filtering: will the model still work in heterogeneous dataset.