CLARE: A Semi-supervised Community Detection Algorithm

Xixi Wu[1], Yun Xiong[1], Yao Zhang[1], Yizhu Jiao[2], Caihua Shan[3], Yiheng Sun[4], Yangyong Zhu[1], and Philip S. Yu[5]

[1]**School of Computer Science, Fudan University** [2]University of Illinois at Urbana-Champaign

[3]Microsoft Research Asia [4]Tencent Weixin Group [5]University of Illinois at Chicago

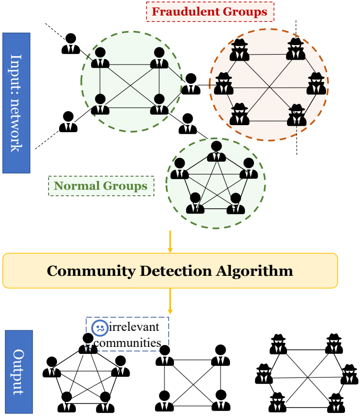
June 22, 2023

28Th ACM
SIGKDD
CONFERENCE
ON KNOWLEDGE DISCOVERY
AND DATA MINING
Washington DC , August 14-18, 2022

Table of Contents

Community Detection

- **Task Definition:** detect subgraphs where nodes are closely related, *i.e.*, communities
- **Drawbacks:** fail to pinpoint a particular kind of community, *i.e.*, targeted community
- **Case:** cannot distinguish fraudulent groups from normal ones in transaction networks

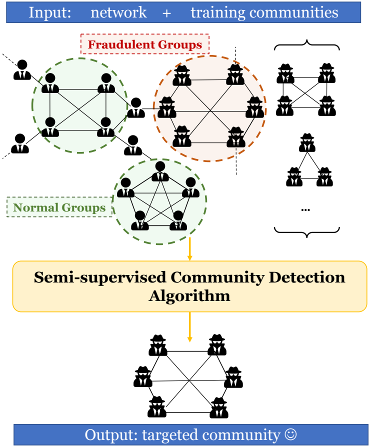


Task Introduction

1 Motivation

Semi-supervised Community Detection

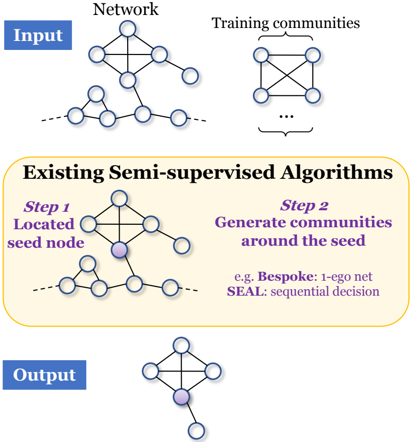
- **Task Definition:** utilize certain communities as training data to recognize the other similar communities in the network
- **Applications:** detect fraud groups in transaction networks; identify social spammer groups in social networks, ...



Existing Methods

Existing methods can be generalized as **seed-based**

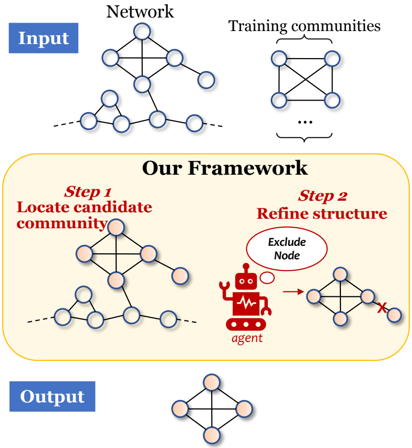
- **Methodology:** *first locate seed nodes (central nodes), then develop communities around seeds*
- **Drawbacks:** quite **sensitive** to the quality of selected seeds :(
  - **Bespoke:** inflexible as returning 1-ego net
  - **SEAL:** time-consuming as generating via sequential decisions



Our Framework

We propose a novel **subgraph-based** inference framework:

- **Methodology:** *first locate candidate communities, then refine their structures*
- **Benefits**
  - More precise positioning (subgraph vs. node)
  - More efficient
  - Further optimization



CLARE Overview

We propose **CLARE** consisting of **C**ommunity **L**ocator **A**nd Community **RE**writer

- Community Locator: locate potential communities by seeking subgraphs that are similar to training ones
- Community Rewriter: refine located communities' structures enhanced by RL

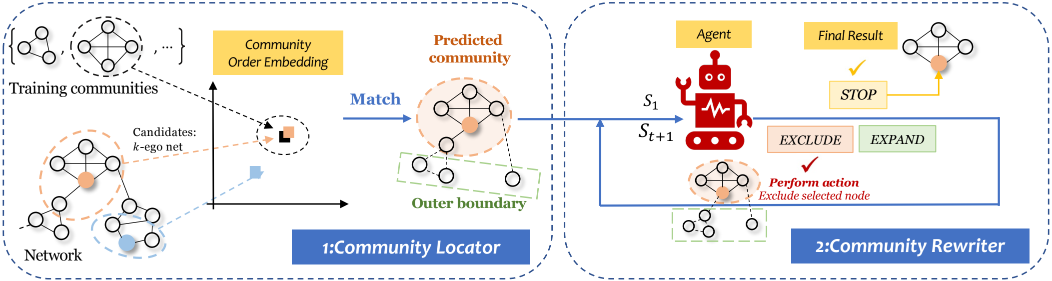


Figure: CLARE framework overview

Table of Contents

Semi-supervised Community Detection

Given a graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges, and $\mathbf{X}$ is the node feature matrix.
With m labeled communities as training data $\dot{\mathcal{C}} = \{\dot{C}^1, \dot{C}^2, ..., \dot{C}^m\}(\forall_{i=1}^{m}\dot{C}^i \subset G)$, our goal is to find the set of other similar communities $\hat{\mathcal{C}}$ in G.

We first encode all training communities and candidate communities, and then locate the potential ones in candidate sets based on similarity.

- **Community Encoder:** For node v, its raw features are $\mathbf{x}(u)$, after k-layers GNN, its final embedding is denoted as $\mathbf{z}(u) \in \mathbb{R}^d$; For a specific community C^i, its embedding is calculated as $z(C^i) = \sum_{v \in C^i} z(v)$.
- **Similarity:** We implement community order embedding: if community C^a is a subgraph of community C^b, then corresponding embedding $\mathbf{z}(C^a)$ has to be in the "lower-left" of $\mathbf{z}(C^b)$: $\mathbf{z}(C^a)[i] \leq \mathbf{z}(C^b)[i], \ \forall_{i=1}^{d}$, iff $C^a \subseteq C^b$. Therefore, the distance of two communities' embedding can be regarded as a measure of similarity.
- **Matching:** Encode training communities as $\dot{\mathbf{Z}} = \{\mathbf{z}(\dot{C}^1), \dots, \mathbf{z}(\dot{C}^m)\}$, candidate communities as $\mathbf{Z} = \{\mathbf{z}(C^1), \ \dots, \ \mathbf{z}(C^{|\mathcal{V}|})\}$ (C^i denotes the k-ego net of node $i \in \mathcal{V}$). Then the n ($n = \frac{N}{m}$) candidate communities **closest to each training one** in the embedding space are considered as predicted results.

In Community Locator, for efficiently locating potential communities, we regard the k-ego net of each node in the network as a candidate community. Such an assumption on the structure of predicted communities is quite inflexible. Therefore, we propose rewriter to intelligently refine their structures.

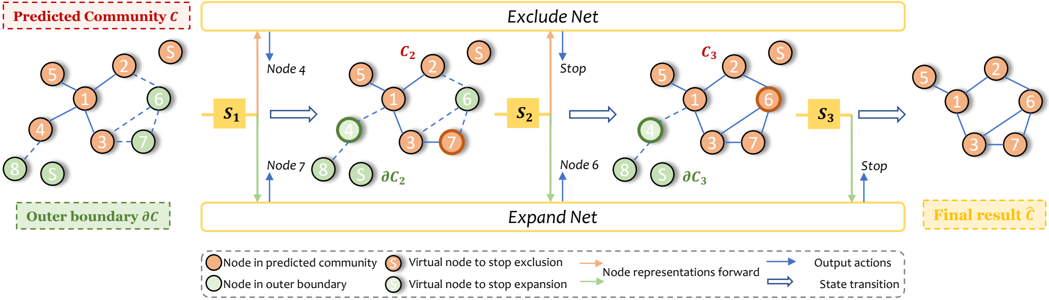


Figure: Illustration of rewriting process

Summary

- Firstly, we train the community locator by leveraging known communities.
- Then we take each training community as a pattern for matching n closest candidate communities in the embedding space ($n = \frac{N}{m}$). Actually, the k-ego net of each node in the network serves as a candidate. After matching, we can get N raw predicted communities.
- Next, we train the community rewriter via policy gradient[1].
- For each community detected in the first stage, it is fed to well-trained agent and refined into a new community.
- Finally, we obtain N modified communities as final results.

[1]For more details, please refer to our original paper

Table of Contents

Experimental Setup

3 Experiments

- **Datasets:**
  - Single datasets: real-world networks containing overlapping communities
  - Hybrid datasets: combination of two different single datasets (by randomly adding cross-network links) to simulate a larger network with different types of communities
- **Baselines:**
  - Community detection methods: BigClam, ComE, CommunityGAN, vGraph
  - Semi-supervised community detection methods: Bespoke and SEAL
- **Evaluation Metrics:** F1, Jaccard, and ONMI

	#N	#E	#C	C_{Max}	C_{Avg}
Amazon	6,926	17,893	1,000	30	9.38
DBLP	37,020	149,501	1,000	16	8.37
Livejournal	69,860	911,179	1,000	30	13.00
Amazon+DBLP	43,946	172,394	2,000	30	8.88
DBLP+Livejournal	106,880	1,070,680	2,000	30	10.69

Overall Performance

3 Experiments

Table 3: Summary of the performance in comparison with baselines. N/A means the model fails to converge in 2 days. We report the results of CLARE with k=1 on DBLP while k=2 on all other datasets.

	Dataset	BigClam	BigClam-A	ComE	CommunityGAN	vGraph	Bespoke	SEAL	CLARE
F1	Amazon	0.6885	0.6562	0.6569	0.6701	0.6895	0.5193	0.7252	**0.7730**
	DBLP	0.3217	0.3242	N/A	0.3541	0.1134	0.2956	0.2914	**0.3835**
	Livejournal	0.3917	0.3934	N/A	0.4067	0.0429	0.1706	0.4638	**0.4950**
	Amazon*DBLP	0.1759	0.1745	N/A	0.0204	0.0769	0.0641	0.2733	**0.3988**
	DBLP*Amazon	0.2363	0.2346	N/A	0.0764	0.1002	0.2464	0.1317	**0.2901**
	DBLP*Livejournal	0.0909	0.0859	N/A	0.0251	0.0131	0.0817	0.1906	**0.2480**
	Livejournal*DBLP	0.2183	0.2139	N/A	0.0142	0.0206	0.1893	0.2291	**0.2894**
Jaccard	Amazon	0.5874	0.5623	0.5691	0.6045	0.5721	0.4415	0.6792	**0.6827**
	DBLP	0.2186	0.2203	N/A	0.2830	0.0645	0.2593	0.2143	**0.3132**
	Livejournal	0.3102	0.3076	N/A	0.3183	0.0222	0.1324	0.3795	**0.4027**
	Amazon*DBLP	0.1102	0.1095	N/A	0.0109	0.0421	0.0488	0.2419	**0.3241**
	DBLP*Amazon	0.1485	0.1478	N/A	0.0610	0.0555	0.2135	0.0879	**0.2166**
	DBLP*Livejournal	0.0523	0.0485	N/A	0.0120	0.0066	0.0756	0.1485	**0.1893**
	Livejournal*DBLP	0.1505	0.1464	N/A	0.0097	0.0105	0.1503	0.1907	**0.2308**
ONMI	Amazon	0.5865	0.5625	0.5570	0.6040	0.5532	0.4129	0.6862	**0.7015**
	DBLP	0.1113	0.1110	N/A	0.2324	0.0020	0.2347	0.1603	**0.2600**
	Livejournal	0.2696	0.2641	N/A	0.3171	<1e-4	0.1024	0.3695	**0.3703**
	Amazon*DBLP	0.0305	0.0334	N/A	<1e-4	< 1e-4	0.0364	0.2475	**0.3126**
	DBLP*Amazon	0.0471	0.0477	N/A	0.0523	<1e-4	**0.1780**	0.0380	0.1566
	DBLP*Livejournal	0.0113	0.0065	N/A	<1e-4	<1e-4	0.0723	0.1155	**0.1331**
	Livejournal*DBLP	0.0858	0.0795	N/A	0.0053	<1e-4	0.1248	0.1906	**0.2012**

Ablation Study

Community Rewriter learns quite different rewriting heuristics for different networks, showing its adaptability and flexibility.

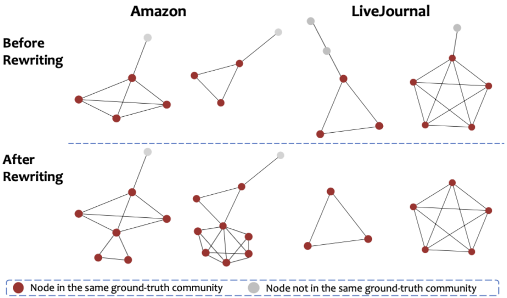


Figure 5: Case study of the community rewriter. On Amazon, many undetected nodes can be correctly absorbed while irrelevant nodes are correctly removed on Livejournal.

- **Paper Title: CLARE: A Semi-supervised Community Detection Algorithm**
- **Code:** https://github.com/FDUDSDE/KDD2022CLARE
- **Contact:** Xixi Wu (xxwu1120@gmail.com / 21210240043@m.fudan.edu.cn)